DETECTION OF BEARING DEFECTS WITH APPROXIMATE BEARING CONFIGURATION

A Dissertation
Presented to
The Academic Faculty

by

Eymard Prevost

In Partial Fulfillment
of the Requirements for the Degree of
MASTER OF SCIENCE in Mechanical Engineering in the
SCHOOL OF GEORGE W. WOODRUFF SCHOOL OF MECHANICAL ENGINEERING

Georgia Institute of Technology
AUGUST 2019

DETECTION OF BEARING DEFECTS WITH APPROXIMATE BEARING CONFIGURATION

Approved by:

Dr. Kurfess, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Saldana
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Fu
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: July 03, 2019

« La routourne va tourner », F. Ribéry

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor, Dr. Thomas Kurfess, who has guided, supported and advised me throughout the year. In his company, I learned a lot, I had the opportunity to discover very interesting projects and research areas. I will never be grateful enough for everything he has done for me. I would also like to thank Dr. Saldana for his support and valuable advice throughout the year, which have enabled me to improve many different skills (communication, methodology...). Thank you also to Dr. Fu for agreeing to serve on this committee and to spend time reviewing this thesis.

Second, I would like to thank our partner, Georgia Pacific company, and in particular Mr. Journeaux, Mr. Ray, Mr. Sylvester, Mr. Burkett, Mr. Peterson, Mr. Coyne for their help and their warm welcome. The year project we did together was exciting and very enriching. Most of the skills I used to do this thesis have been learned / improved during this project, helping me to complete it in an efficient way.

Third, my thanks also go to the administrative staff of the Office of International Education, and the School of Mechanical Engineering, especially Mrs. Glenda Johnson whose explanation tremendously helped me with paperwork.

Ultimately, I would like to thank the entire team Kurfess, in particular Niklas, Pierrick, Daniel, Sam, Kyle, Mahmoud and Dong Min for their unfailing support, their wise advice and with whom I developed a genuine relation of friendship.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

RPM	Rotations per minute
BPFO	Outer race defect frequency
BPFI	Inner race defect frequency
BSF	Ball defect frequency
FTF	Cage defect frequency
IoT	Internet of Things
AI	Artificial intelligence
Dball	Ball diameter
Dpitch	Pitch diameter
FFT	Fast Fourier transform

SUMMARY

Unscheduled maintenance in a production line due to breakdowns is highly detrimental to production operations and profitability. The ability to predict impending failure and replace components before they fail during regularly scheduled maintenance is a high value proposition, as this approach avoids unscheduled operational shutdowns. Such a prediction can be achieved by monitoring key signatures in components that are known to fail often in mechanical systems, such as bearings. Prior research has led to the development of bearing monitoring approaches that are widely employed on industrial systems. However, one of the main challenges that face such modern diagnostic and prognostic systems is the fact that there is often incomplete information about the systems available to the diagnostic tools. For example, rotational velocity and bearing configuration are critical pieces of information that are needed for the diagnostic algorithms to function properly; yet, rotational velocity is typically known to only 10% or 20% of its actual value. Furthermore, previous maintenance operations may have replaced rolling element bearings with bearings that are a slightly different configuration than the original equipment (e.g., a different number of rolling elements). This thesis will focus on approaches that can be employed to detect bearing defects and incipient bearing failure in the presence of incomplete and inaccurate system knowledge.

CHAPTER 1: INTRODUCTION

Throughout history, the industrial domain has experienced different revolutions. During the 18th century, the use of steam led to mechanization. At the end of the 19th century, the use of electricity and the division of labor triggered the second revolution, resulting in a mass production. The third one benefited from the progress of electronics and information technologies to automize simple tasks. Thanks to the progress made simultaneously in the numerical and digital domains, the industry is now experiencing its 4th industrial revolution. As explained in [1, 2], this revolution has resulted in the collaboration of sectors that used to be quite independent from one another (since they had not reached enough maturity). These include artificial intelligence, robotics, the Internet of Things, additive manufacturing (3D printing), genetic engineering, and virtual reality. The goal now is to integrate all improvements made in those sectors to enhance productivity, profits while reducing the carbon footprint, thanks to a more efficient use of resources.

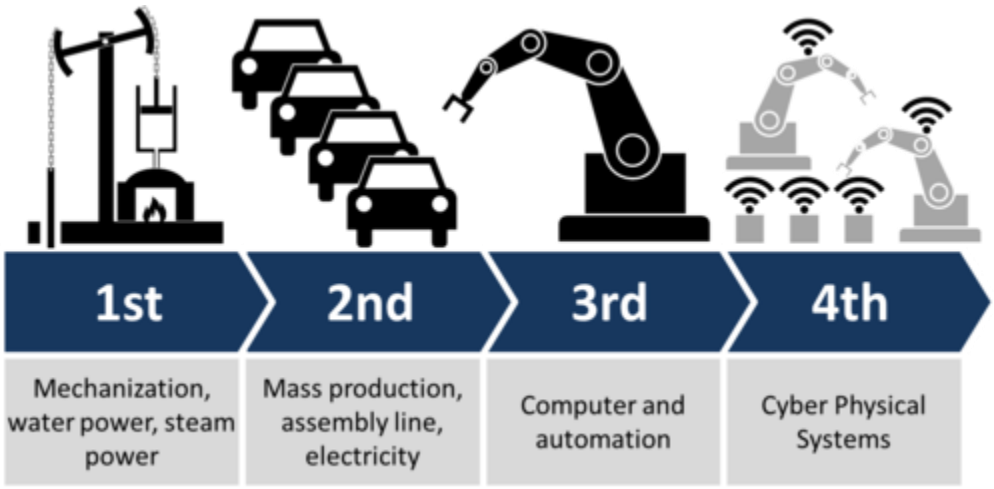


Figure 1 – The different industrial revolutions and their causes

Another cause that triggered the 4th revolution is the vast amount of data that is now available due to low price and more precise sensors, to increased storage capabilities / development of competitive cloud solutions etc. The data driven approach is henceforth a pillar of this revolution. It brings new challenges: for various reasons, the data quality acquired and stored is sometimes too low. Choosing which parameters to collect is sometimes a challenging task. The way it is stored is another issue many companies face. But once those barriers are overcome, powerful enhancements are made possible, among them, machine health monitoring. Thanks to key parameters and to scientific studies that were done few decades ago, industries have at their disposal tools that can be used to determine the health of a part of a machine. These tools can then be exploited to anticipate failures, realize maintenance operations if required, and thus, avoid the production line stopping due to a part break.

This thesis will emphasize bearing health monitoring. As said previously, physical models have been developed, but some blocking points still exist. Due to inaccurate

parameter values, the automated systems aren't able to realize a reliable diagnostic. As a matter of consequence, we will focus on approaches that can be employed to detect bearing defects and incipient bearing failure in the presence of incomplete and inaccurate system knowledge.

CHAPTER 2: THEORETICAL BACKGROUND

2.1 Machine Health Monitoring

Companies use different maintenance strategies [3]. Some are relatively straight forward to implement, others more difficult. The advantage of implementing complex strategies is that in the long run it can be less expensive according to [4].

2.1.1 Maintenance strategies

1. **Run-to-break**: the machine runs until it fails. The advantage is that it does not require any expensive monitoring and one gets longer times between shutdowns. But when the failure occurs, it is unpredicted and can cause important disturbance, in the worst case stopping the production line. This can be very costly, and often, more expensive than the machine itself. Moreover, when the break down happens, some other components may be damaged, increasing the cost and time of repair. Thus, this kind of maintenance is often avoided, except in industries where the loss of machines is not critical [4].
2. **Time based maintenance**: maintenance is realized with a certain regularity, in order to get an unpredicted failure rate below a certain threshold (around 1-2% in many cases). The failures are much less frequent, but still occur. Besides, this generates a consequent waste and extra spending, as most of the equipment could in fact work for two or three times the maintaince period chosen [5]. Good parts may be replaced even though they are still functional

. A side effect is the impact this waste produces on workers, who may feel they spend most of their time doing unnecessary replacements, impacting the maintenance quality [4]. It has been observed in some cases that this maintenance strategy even produced a rise in very early failures, the replacement being imperfect [6]. This approach still remains relevant in cases where the failure rate can be predicted with high confidence [4].

3. **Predictive maintenance**: also named condition based maintenance, this is often the most efficient method, in terms of cost productivity but also the most challenging [4, 7]. The idea is to rely on key parameters to determine the machine health and, when reaching a danger zone, effecting a replacement. According to Bloch and Geitner in [8], this philosophy relies on the observation that the vast majority of equipment failures are preceded with signs. Predictive maintenance has gained a lot of popularity in past decades [9], as the range of monitoring techniques increased very much and as data availability is gets easier. Nonetheless, many challenges remain. The data acquisition chain is one: it is sometimes unreliable, unsecure, too slow [10]. The engineers have to face data inconsistency, incompleteness [9] due to multiple reasons. Besides, in some cases, the environment is complex, in the sense that part of the data required to realize efficient diagnosis and prediction is missing. This can explain why though many physical models have been discovered more than 20 years ago (at least for the diagnostic part), industries sometime struggle to implement efficient predictive maintenance.

2.1.2 Predictive maintenance methods

Predictive maintenance relies on the analysis of key parameters in order to establish a diagnostic or a prediction. Many different types of parameters are monitored, depending on the machine studied. This leads to different fields of study, among them:

2.1.2.1 Vibration analysis

Every part in movement is emitting characteristic vibrations. As its health changes, the signature evolves in a predictable way. Therefore, it can be used to establish a diagnostic. Vibration analysis is one of the most powerful diagnostic technics and has been the subject of numerous scientific studies [11].

2.1.2.2 Acoustic analysis

Instead of relying on waves propagating through solids, one can consider studying acoustics waves propagating in air. The frequencies studied can be relatively high (for metallic elements it is around 100kHz to 1MHz [4]). The approach has shown a certain success for bearings in some applications [12] but this isn not systematic [13]. Besides it can be complex to implement, and the high sampling rate requires important storage capacities [4].

2.1.2.3 Lubricant analysis

The lubricant transports information from the machines by carrying wear particles, chemical contaminants, etc. Its use is mainly limited to oil circulation lubrication systems and sometimes to grease lubricants.

2.1.2.4 Visual analysis

This is one of the most used and intuitive approaches. Because of the progress of electronics and the decreasing cost of electronic device, visuals analysis has seen significant growth since the 80s. Thanks to the improvements made in artificial intelligence and especially deep learning, this sector faces an important boom.

Since this thesis is in the context of industry 4.0, the predictive maintenance approach is chosen. To monitor equipment health, vibration analysis is used as it is very common in industry and still benefits from state-of-the-art improvements.

2.2 Bearing

2.2.1 Bearing components

The bearings are some of the most prominent components used in industry. Those devices are used to guide a rotating assembly, i.e. allow a part to rotate around a defined axis. It is composed of:

- Rolling elements. Those can be spheres, cylinders, cones etc. According to the direction of the load and its magnitude, one type is preferred to the others
- Inner and outer race. Parts between the rolling elements move. Usually, the outer race is motionless in the reference frame and the inner race is turning (this is important to mention as it has an impact for the rest of the thesis)
- Cage. Element containing the rolling elements. It prevents the latter to rub one against another and thus reduces friction and wear.

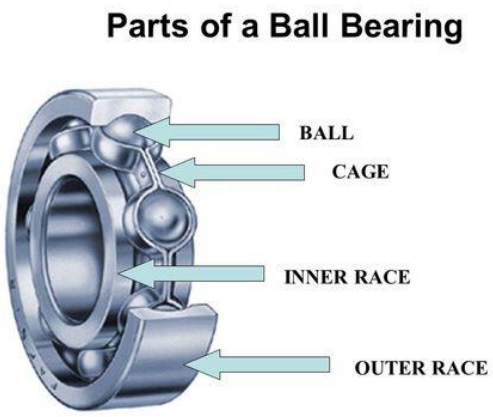


Figure 2 – Principal components of a ball bearing

2.2.2 Bearing geometry

Let's introduce some geometrical parameters of bearings which are going to be important for the rest of the document:

- Rolling element diameter (D_{ball} on Figure 3)
- Contact angle. Defined as the angle made by the line linking the points of contact of the ball and the inner and outer raceway ($\boldsymbol{\phi}$ on Figure 3)
- Pitch diameter. Diameter of the circle made in one rotation by the center point of the rolling element (D_{pitch} on Figure 3)

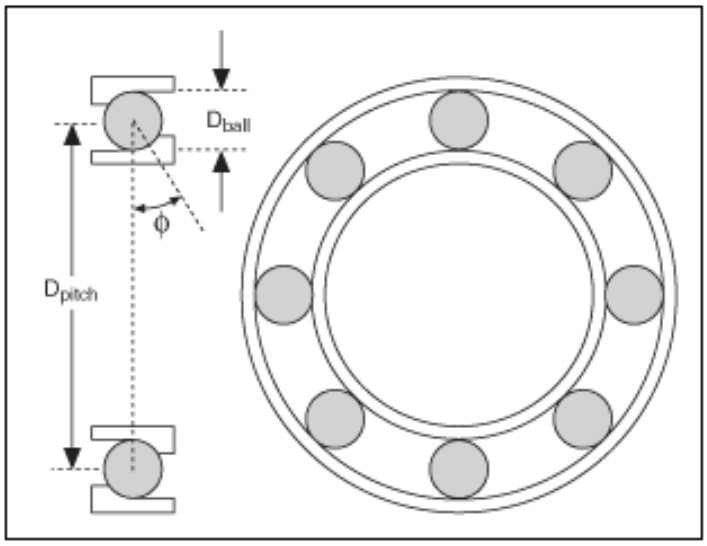


Figure 3 – Roller bearing important geometric parameters

2.2.3 Bearing defects

Like any mechanical component, bearings are put under stress conditions. The latter causes the bearing to fail at a certain point. Multiple different kinds of failures exist, among them:

- Outer and inner race defect: it appears when the race is damaged and has spall and cracks
- Rolling element defect: the surface of the latter is damaged
- Cage defect: less frequent than the other defects mentioned, as the cage is usually under a smaller load

Those equations are used later in this thesis to generate data, and to get an approximation of the defect frequency.

2.3 Vibration analysis

Different methods exist to monitor the bearing health. But one of the most efficient is to rely on vibration analysis. Indeed, when cracks form in bearings, additional vibrations appear at some defined frequencies. First, we introduce the mathematical background and second, we apply it to the bearing case.

2.3.1 Fourier transform

This is one of the most important tools that vibration analysis has. It takes as an input a temporal signal and gives its frequency decomposition, from which one usually gets much more insight.

If *f* is an integrable function on $\mathbb{R}$, then its Fourier transform is the function $F : \mathbb{R} \rightarrow \mathbb{C}$ defined by:

$$\boldsymbol{F(\xi) = \int_{-\infty}^{+\infty} f(x) * e^{-2i\pi x\xi} dx} \qquad \textbf{(1)}$$

Where ξ is a real number

The transformation can be reversed to get f back:

$$\boldsymbol{f(x) = \int_{-\infty}^{+\infty} F(\xi) * e^{2i\pi x\xi} d\xi} \qquad \textbf{(2)}$$

As we are working in a real world, one must adapt the equations to get a definition working for discrete cases. Let's assume this time *f* is a discrete function composed of n elements: $\{f_0, f_1, \ldots, f_{n-1}\}$. Then, the discrete Fourier transform will be composed of n complex elements $\{F_0, F_1, \ldots, F_{n-1}\}$ given by:

$$F_k = \sum_{j=0}^{n-1} f_j * e^{\frac{-2i\pi}{n} * k * j} \quad (3)$$

The function *f* is obtained back by using the following formula:

$$f_k = \frac{1}{n} \sum_{j=0}^{n-1} F_i * e^{\frac{2i\pi}{n} * k * j} \quad (4)$$

2.3.1.1 Fast Fourier transform

A bottleneck with the discrete Fourier transform (DFT) is its computational cost: for n outputs F_k, there are n operations involved. This gives a $O(n^2)$ complexity. Quadratic complexity can be an important issue when working with large quantity of data. Optimizations have been proposed, amongst them, the fast Fourier transform (FFT) by Cooley and Tukey [14]. Its complexity is in $O(n * \log(n))$ [15]. The FFT requires the number of points to be a 2^n value to obtain best improvements. Some generalisations allow this number to be different, but the speed increase gets reduced.

Different versions of this algorithm exist. One of the most popular is the Cooley-Tukey version, more specifically the radix-2 decimation case. The idea is to divide the DFT into 2 equal parts recursively [14, 15]. The DFT formula is:

$$F_k = \sum_{j=0}^{n-1} f_j * e^{\frac{-2i\pi}{n} * k * j} \quad (5)$$

with *k* an integer ranging from 0 to *N – 1*. Then one splits the indexes into the odd and even numbers:

$$F_k = \sum_{j=0}^{\frac{n}{2}-1} f_{2j} * e^{\frac{-2i\pi}{n} * k * 2j} + \sum_{j=0}^{\frac{n}{2}-1} f_{2j+1} * e^{\frac{-2i\pi}{n} * k * (2j+1)} \quad (6)$$

Then we extract the term $e^{\frac{-2i\pi}{n}}$ from the right part. Cooley and Tukey have remarked that the left part represents the DFT of the even-indexed inputs, and the right part the odd-indexed inputs:

$$\boldsymbol{F_k = \sum_{j=0}^{\frac{n}{2}-1} f_{2j} * e^{\frac{-2i\pi}{n} * k * 2j} + e^{\frac{-2i\pi}{n} * k} * \sum_{j=0}^{\frac{n}{2}-1} f_{2j+1} * e^{\frac{-2i\pi}{n} * k * 2j}} \quad \textbf{(7)}$$

By renaming the left even part E_k and the right odd part O_k we obtain:

$$\boldsymbol{F_k = E_k + e^{\frac{-2i\pi}{n} * k} * O_k} \quad \textbf{(8)}$$

It also has been found that $F_{k\,+\,n/2}$ can be calculated using E_k and O_k. This can be used to reduce the number of operations to implement.

$$\boldsymbol{F_{k+\frac{n}{2}} = \sum_{j=0}^{\frac{n}{2}-1} f_{2j} * e^{\frac{-2i\pi}{n} * (k+\frac{n}{2}) * 2j} + e^{\frac{-2i\pi}{n} * (k+\frac{n}{2})} * \sum_{j=0}^{\frac{n}{2}-1} f_{2j+1} * e^{\frac{-2i\pi}{n} * (k+\frac{n}{2}) * 2j}}$$

$$F_{k+\frac{n}{2}} = \sum_{j=0}^{\frac{n}{2}-1} f_{2j} * e^{\frac{-2i\pi}{n} * k * 2j} e^{-2i\pi j} + e^{\frac{-2i\pi}{n} k} e^{-i\pi} * \sum_{j=0}^{\frac{n}{2}-1} f_{2j+1} * e^{\frac{-2i\pi}{n} k} e^{-2i\pi j}$$

$$\boldsymbol{F_{k+\frac{n}{2}} = \sum_{j=0}^{\frac{n}{2}-1} f_{2j} * e^{\frac{-2i\pi}{n} * k * 2j} - e^{\frac{-2i\pi}{n} k} * \sum_{j=0}^{\frac{n}{2}-1} f_{2j+1} * e^{\frac{-2i\pi}{n} k}}$$

$$\boldsymbol{F_{k+\frac{n}{2}} = E_k - e^{\frac{-2i\pi}{n} * k} * O_k} \quad \textbf{(9)}$$

(8) and (9) are used to speed up calculation

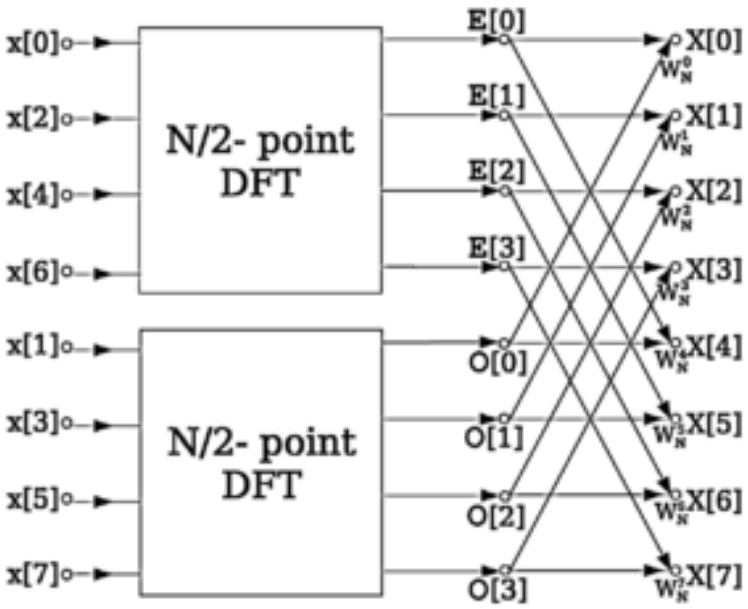


Figure 4 – data flow for an 8-points Fourier transform

2.3.1.2 Hanning window

The mathematical equations are defined on infinite segments. This is not applicable to real cases where we have a limited number of datapoints. To get around the problem, the signal is repeated n times to get a satisfactory number of elements. This brings another issue: the first and last values may not match when reassembling the waveform. A discontinuity appears, causing a spectral leakage (new and artificial frequency components appearing [16]). More precisely, some high frequency components are created (sharp discontinuities are represented by high frequency bins and overall trend by low ones). To reduce this phenomenon a window function is applied. It diminishes the difference between the 2 extrema values thus downsizing the spectra leakage. The most common one is the Hann window [17]:

$$\boldsymbol{H(x) = \frac{1}{2} + \frac{1}{2} * \cos(\Omega * x) = cos^2(\Omega * x)} \textbf{ (10)}$$

where $\Omega = \frac{2*\pi}{numberPoints}$

It viewed as a "raised" cosine, i.e. its minimum being 0 and maximum 1.

2.3.2 Bearing defect frequencies

When a defect starts to form on a bearing, some additional vibrations appear on the spectra. Those vibrations are defined by the location of the defect, the speed and the bearing geometry. Nonetheless, due to slippage and axial loading, those frequencies do not always exactly coincide with the theoretical values. The following equations are valid under certain conditions:

1. Neglection of centrifugal forces or gyroscopic coupling
2. Stationary outer race and rotating inner race
3. The rolling elements does not slip

$$\boldsymbol{bpfi = Z * \frac{f}{2} * \left(1 + \frac{D}{d_m} cos(\alpha)\right)} \quad \textbf{(11)}$$

$$\boldsymbol{bpfo = Z * \frac{f}{2} * \left(1 - \frac{D}{d_m} cos(\alpha)\right)} \quad \textbf{(12)}$$

$$\boldsymbol{fr = Z \frac{f * d_m}{D} * \left(1 - \left(\frac{D}{d_m} cos(\alpha)\right)^2\right)} \quad \textbf{(13)}$$

$$\boldsymbol{ftf = \frac{f}{2} * \left(1 - \frac{D}{d_m} * cos(\alpha)\right)} \quad \textbf{(14)}$$

with:

- *Z*: number of rolling elements
- *f*: RPM frequency (number of rotations per minute divided by 60)

- D: rolling element diameter
- d_m: pitch diameter
- *BPFI*: ball pass frequency inner (race). This frequency (and its harmonics) appears on the spectrum when a defect starts on the inner race
- *BPFO*: ball pass frequency outer (race). This frequency (and its harmonics) appears on the spectrum when a defect starts on the outer race
- f_r: frequency rolling (element). This frequency (and its harmonics) appears on the spectrum when a defect starts on one of the rolling elements
- *FTF*: fundamental train frequency. This frequency (and its harmonics) appears on the spectrum when a defect starts on the cage

In some cases, parameters that are needed to compute the exact defect value are missing. As described in [18], it is possible to get estimates of those frequencies:

$$\boldsymbol{bpfo = 0.4 * f * Z} \quad \textbf{(15)}$$

$$\boldsymbol{bpfi = 0.6 * f * Z} \quad \textbf{(16)}$$

$$\boldsymbol{ftf = 0.4 * f} \quad \textbf{(17)}$$

This approximation can be explained by the fact that from an experimental standpoint, the *FTF* value is usually comprised between 0.33 and 0.50 of the shaft frequency. For most bearings it will be between 0.38 and 0.42 (thus one takes the average and gets the explanation of (17). This means:

$$\boldsymbol{\frac{D}{d_m}\cos(\alpha) = 0.2} \quad \textbf{(18)}$$

Using 13, 14 and 20 we can infer 18 and 19.

2.3.3 Defect spectra

Due to a different geometric configuration, the spectrum of a defect is different, according to its location on the bearing

2.3.3.1 Outer race defect spectra

As we assumed previously, the outer race is fixed in the world frame. Thus, the defect spectrum is composed of a fundamental frequency (which frequency is equal to the BPFO frequency), and harmonic frequencies (multiples of the BPFO frequency).

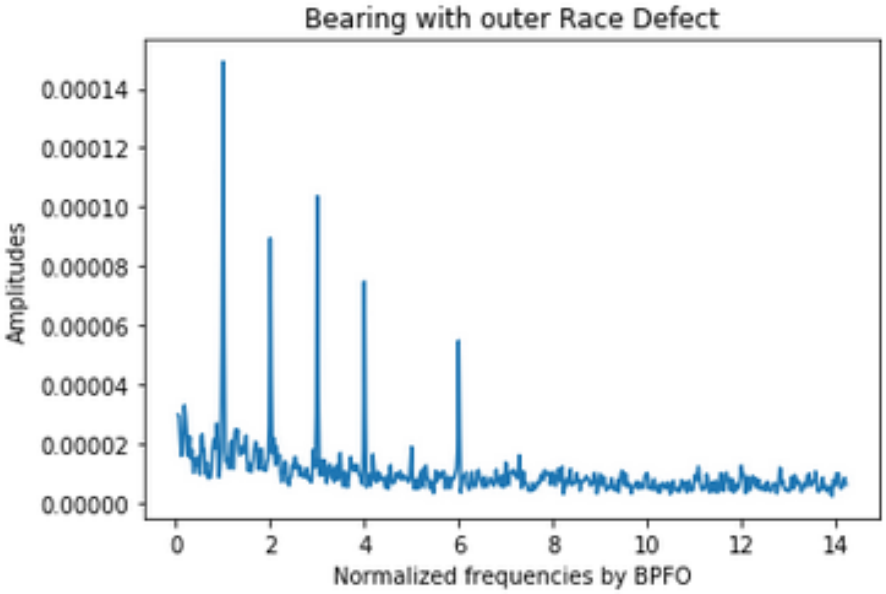


Figure 5 – BPFO spectrum

2.3.3.2 Inner race defect spectra

On the opposite side, the inner race is turning in the world frame. Thus, along the fundamental (=BPFI) and the harmonics appear sidebands. This phenomenon occurs when

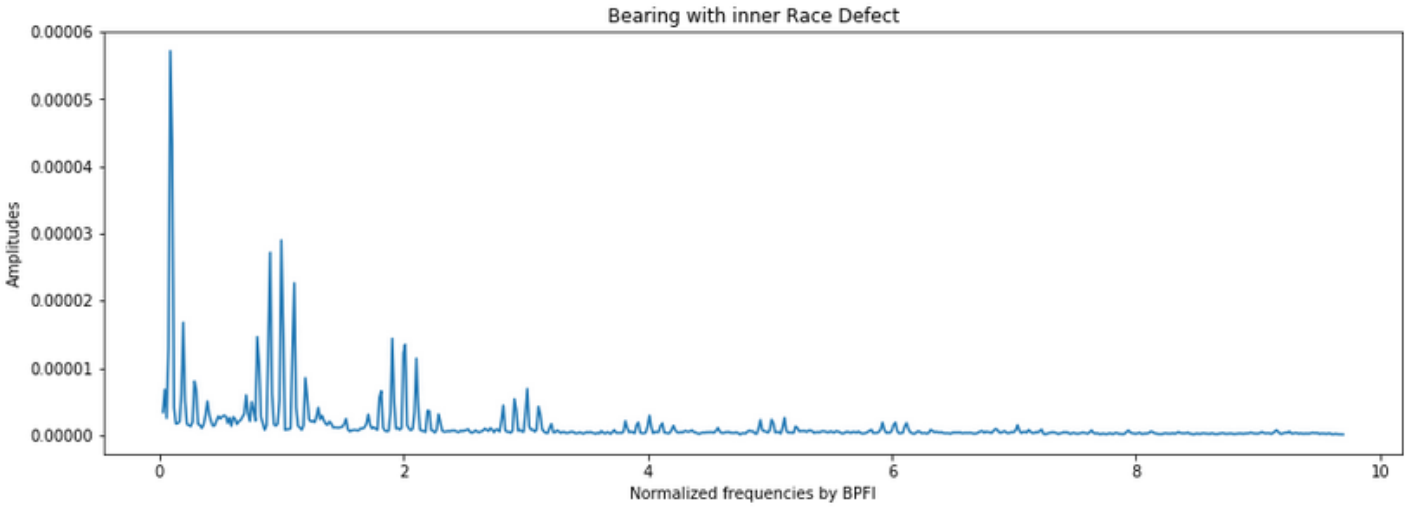


Figure 6 – BPFI spectrum

a frequency (called carrier frequency) is modulated by another one. In the case of BPFI, the rotation frequency is modulating a defect signal. Indeed, the load is localized only at a certain area. Therefore, when the defect of the inner cage is being hit by a rolling element in this zone, the vibration has an important amplitude. On the other hand, when the defect is located on the opposite side of the load and is hit by a ball, the vibration amplitude is minimum [4]. This explains the waveform of Figure 7.

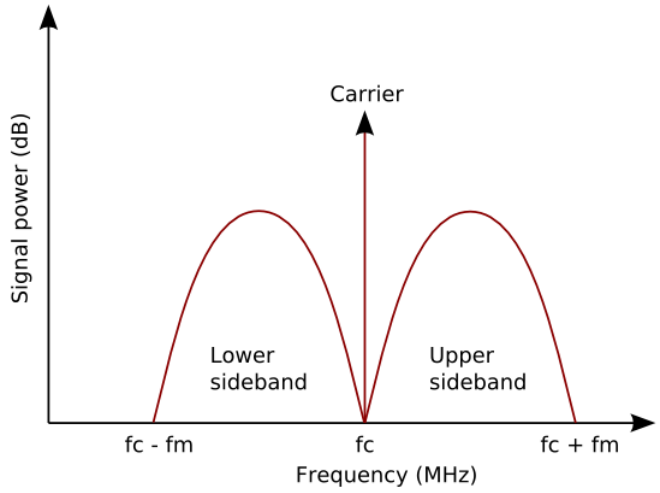


Figure 7 – Amplitude modulation

It can be shown mathematically on the resulting spectrum, the fundamental and harmonics are surrounded by smaller and symmetric peaks.

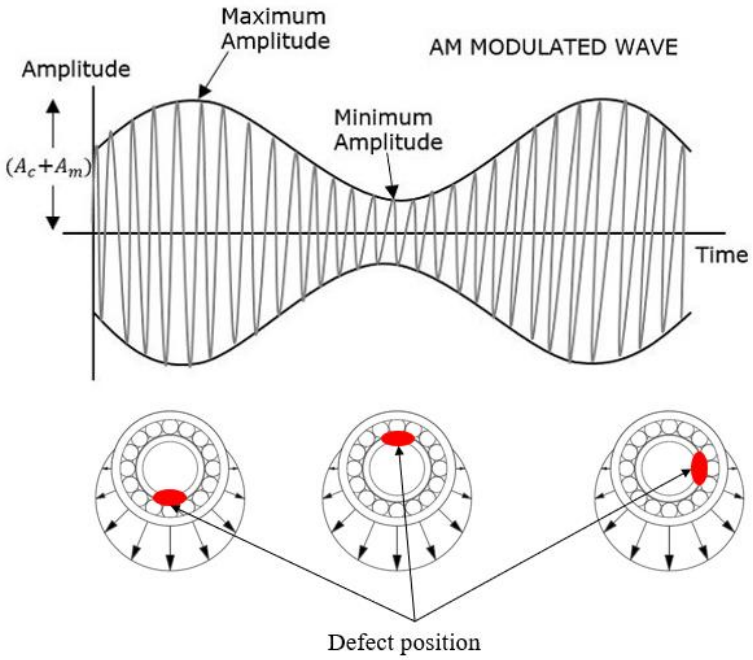


Figure 8 – Resulting spectrum of a modulated waveform (fm: modulating frequency, fc: carrier frequency)

2.3.3.3 Rolling element defect spectra

Again, there is amplitude modulation, but the modulating frequency is the *FTF* (a rolling element completes *FTF* complete rotations per second).

2.3.4 Time domain techniques

Frequential analysis is a very powerful tool at the disposal of the engineer. But it can be more complex compared to other more intuitive and straightforward approaches. This is especially the case when automating a process. Humans are able to derive easily information from a spectrum but in the case of algorithms this is much more difficult. Indeed, there are many different cases to take into account, from a very noisy to a smooth spectrum, from a healthy bearing spectrum to a multiple defect bearing spectra etc. A way to avoid spending a consistent amount of time and money is to use the time domain techniques. This is usually straight forward, brings less insights, but can still be reliable provided that the chain acquisition of the signal is not flawed (defective sensors, unreliable data transmission, etc.). To be efficient, the time domain techniques have to be used in a context of time monitoring (checking the evolution of the parameters), as alone, the figures derived do not make sense.

2.3.4.1 Mean

The mean acceleration is the standard statistical mean value. As it increases, the bearing's health worsens. Let X bet a signal of N elements *xi*. The signal mean is:

$$\bar{x} = \frac{1}{N} * \sum_{i=1}^{N} xi \qquad \textbf{(19)}$$

2.3.4.1 Root Mean Square (RMS)

It describes the overall energy of the signal:

$$\boldsymbol{RMS(X)} = \sqrt{\frac{\sum_{i=1}^{N} xi^2}{N}} \quad \textbf{(20)}$$

This is a powerful tool to detect anomalies. Usually, the system has in its database a reference for the RMS value and thresholds it must not go above or under. The main drawback of this variable is that it can't be used to tell what type of anomaly is present on the bearing. Besides, it is very sensitive to unplanned situations: for instance, an element hitting the machine can make RMS increase by too much, triggering an alarm. Nonetheless, its simplicity makes it attractive, and researchers like Tandon (1994) [19] have used this factor in a successful way.

The RMS can be applied on both the waveform and spectrum.

2.3.4.2 Peak Value

This is applied to the time signal data. It consists of the maximum (absolute) amplitude of the acceleration signal. Tandon in [20] showed that as the bearing defect diameter increases, the peak value gets higher and higher.

2.3.4.3 Skew

This is the third of the standardized moments (the mean is the first moment and the variance is the second). It measures the asymmetry of a signal. If nonzero, it means the latter is unbalanced. It can be used to monitor the evolution of the spectra. As the defect propagates, new frequency components appear while others tend to increase in terms of

amplitude causing the skew to evolve. A too large evolution from the original value should be suspicious and trigger further investigations.

$$\boldsymbol{Skew(X) = \frac{1}{N-1}\sum_{i=1}^{N}(xi - \bar{x})^3} \qquad \textbf{(21)}$$

Interpretation: if the signal is balanced, then the term $(xi - \bar{x})$ should remain very close from 0 as the xi are for most of them around $\bar{x}$. Whereas if it is very asymmetrical (meaning a consequent number of xi are both far away from the mean and on the same side of it) then $(xi - \bar{x})$ is going to be relatively different from 0. Papers have shown it is a relevant parameter to monitor to detect failure [21, 22].

2.3.4.4 Kurtosis

It is defined as being the ratio of the fourth moment to the second moment (variance). As skew coefficient, it gives indication on signal shape, here the flattening of a signal. A smooth surface has a theoretical value of 3 (assuming the surface milling/turning gives a repartition following a normal distribution [23]). As it gets rough, the value increases [23].

$$\boldsymbol{Kurtosis(X) = \frac{(N-1)*\sum_{i=1}^{N}(xi-\bar{x})^4}{(\sum_{i=1}^{N}(xi-\bar{x})^2)^2}} \quad \textbf{(22)}$$

2.3.5 Control charts

To monitor the evolution of time domain parameters, one powerful and common tool is the control chart. Basically, it displays the evolution of a parameter over time and determines if the overall variation corresponds to a controlled process or not.

2.3.5.1 X bar chart

X bar charts are used to monitor the mean of a parameter. Samples are gathered in subgroups (usually smaller than 10 elements as detailed in [24]). The overall mean and standard deviation of all the subgroups is computed. One can draw boundaries from the standard deviation: assuming the process is following a normal probability distribution law then 99.7% of the subgroups should be contained between $mean - 3 * standard\ deviation$ and $mean + 3 * standard\ deviation$. A subgroup going outside of those limits should raise a flag. If the next points come back into the boundaries, it can then be assumed we were in the 0.3% percent cases where the values are outside of the limits computed. If not, then an anomaly is going on: assuming the distribution law this is very unlikely to happen.

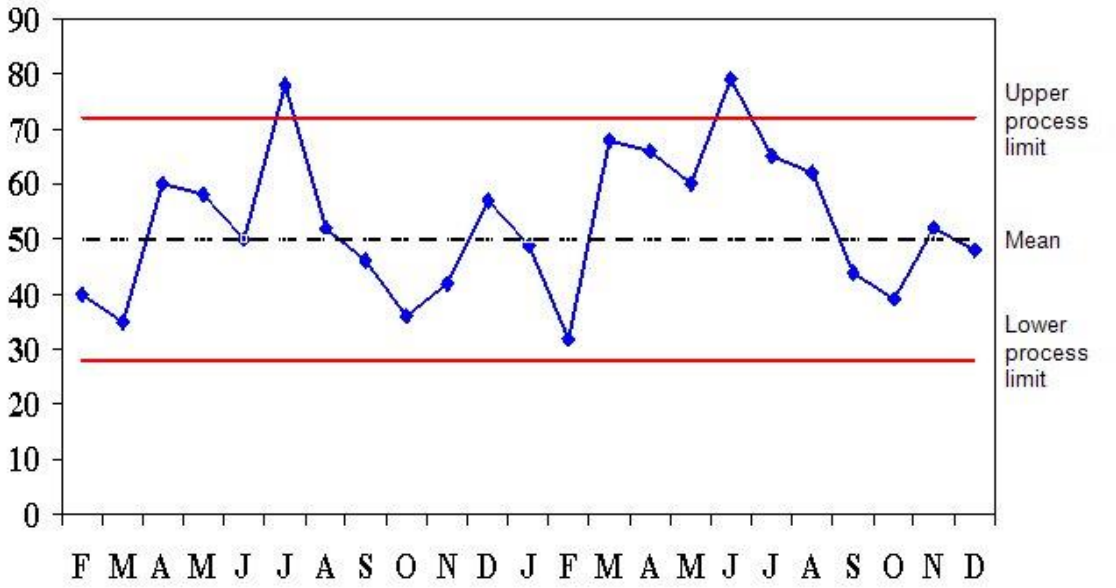


Figure 9 – Mean control chart over months

2.3.5.2 S bar chart

The S bar chart is used to monitor the evolution of the standard deviation of the process. The samples are gathered in subgroups and the standard deviation (s) is computed. The upper and lower control limits (UCL and LCL) are calculated following the formula [25, 26]:

$$LCL = \bar{s} - 3 * \frac{\bar{s}}{c4}\sqrt{1 - c4^2} \qquad (23)$$

$$center\ line = \bar{s}$$

$$UCL = \bar{s} + 3 * \frac{\bar{s}}{c4}\sqrt{1 - c4^2} \qquad (24)$$

The *c4* factor is a constant depending on the number of samples (n) in subgroups:

$$s4 = \sqrt{\frac{2}{n-1}} * \frac{\left(\frac{n}{2}-1\right)!}{\left(\frac{n-1}{2}-1\right)!} \qquad (25)$$

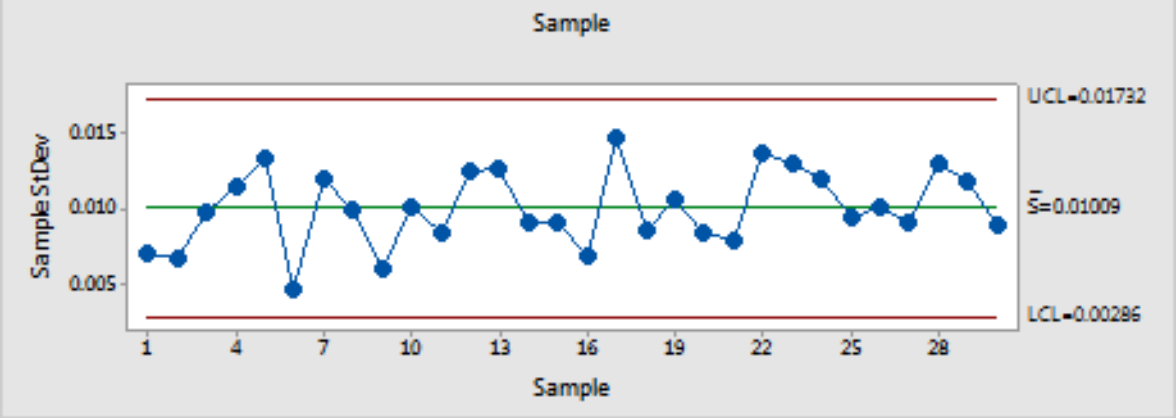
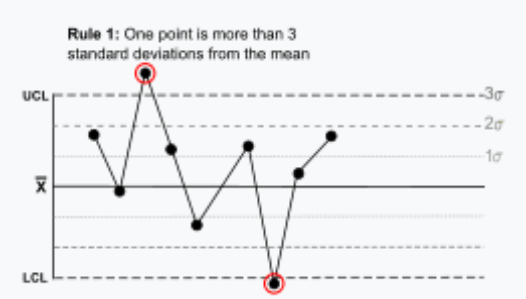
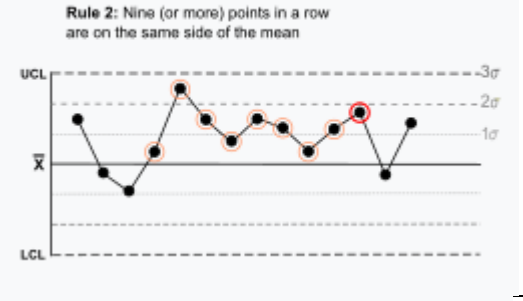
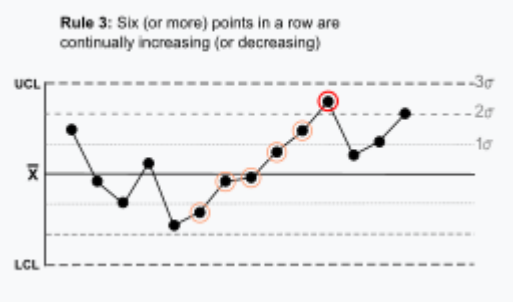
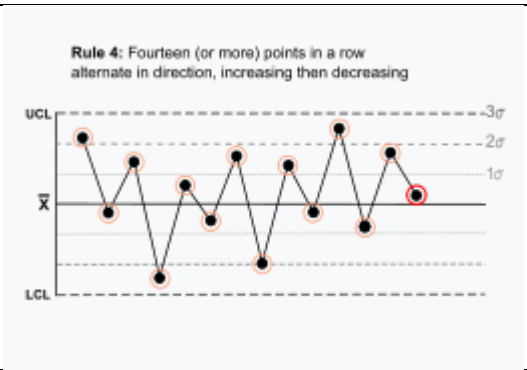
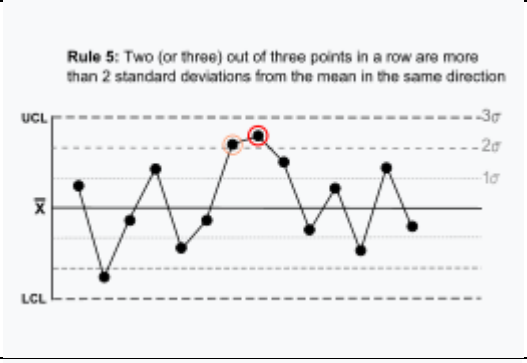
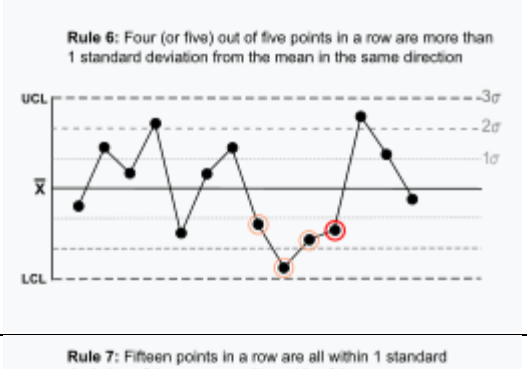
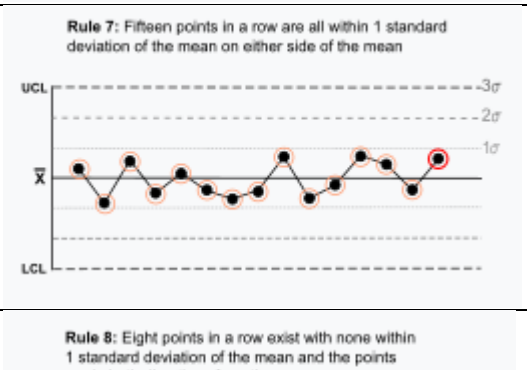
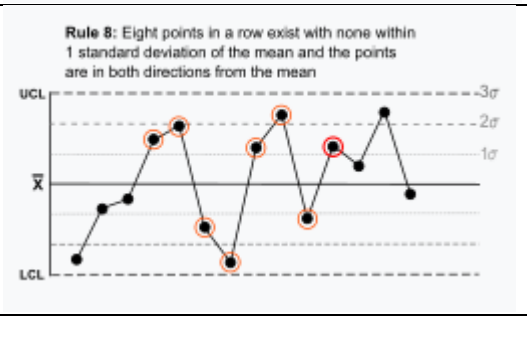


Figure 10 – S chart

2.3.5.3 Nelson rules

Nelson rules were first presented in [27]. If one assumes the process is following a normal probability distribution law, then other interesting patterns can be used to flag an anomaly (and hopefully bearing defects). From [28], we know that all those patterns have a probability of less than a 1% of appearing:

Rule	Example	Probability of happening
Rule 1: one point away from 3 standard deviations		0.0027
Rule 2: at least 9 points in a row are on the same side of the mean		0.00391
Rule 3: at least 6 points in a row a continually increasing/decreasing		0.00278

Rule 4: at least 14 points in a row alternate (increase then decrease)		0.00457
Rule 5: 2 or 3 points out of 3 are on 2 standard deviations away from the mean and on the same side		0.00306
Rule 6: 4 points out of 5 are 1 standard deviation from the mean on the same side		0.00553
Rule 7: at least 15 points are 1 standard deviation away from the mean (side does not matter)		0.00326
Rule 8: at least 8 points in a row are away from more than 1 standard deviation from the mean (side does not matter)		0.0001

2.3.6 Cepstrum

Cepstrum was first introduced in 1963 by Tukey et al. in [29]. Two different definitions of the cepstrum exist [30]. The first one defines it as being the Fourier transform of the natural logarithm applied to the Fourier transform of a waveform. The second (more current) defines it as being the Fourier transform of the inverse Fourier transform of the natural logarithm applied to the waveform signal. It has been applied successfully in gear analysis [31], as it groups the sidebands into a single signature, which eases the automation of detection and interpretation. In bearing fault detection, it has been proven Cepstrum is more efficient with high speed turning machines. Cepstrum application is not always successful, as emphasized by Tandon in [18, 19]. He was able to detect BPFO but not BPFI.

2.4 Optimisation methods

When working on an engineering problem, in many cases one does not have the entire equations describing the system, or, there are no analytical solutions. This is regularly the case when working with partial derivative equations, which are basically everywhere in physical world. To find the best solution, one has thus to test different parameter combinations to get the best possible result. Different methods for that exist. Some are straightforward to understand and implement, others are the opposite, but are usually more efficient in terms of speed and output result.

2.4.1 Zero order algorithms

Those algorithms are the simplest to implement but are usually inefficient in terms of calculation and output result compared to more sophisticated methods. Nonetheless, they

can sometimes be used in contexts where the higher order algorithms can't be used, due to a too big number of local minimums and maximums. It has been discovered that it can be quite efficient in some contexts when coupled with a 1st order algorithm [32]. The most famous is the random search. The latter consists of choosing randomly the parameters, then evaluating the output result until the score is satisfactory.

2.4.2 First order optimization algorithms

2.4.2.1 Explanations

Explanations rely on the first derivatives of the parameters (also called the gradient). Indeed, the fact that they are equal to 0 when reaching a maximum or a minimum can be exploited. Besides, the gradient has the advantage of pointing in the direction of the biggest increase [33], thus taking the opposite direction gives us the biggest decay (assuming the function is defined and differentiable). By repeating this process, hopefully, one should reach the global minimum.

Method: let's assume we have a multivariable function F, which is defined and differentiable for any real value of those parameters. Then as described in [34]:

1. Compute the value of a random point a_n
2. Get the gradient ∇F of F at point a_n
3. Compute $a_{n+1} = a_n - \gamma \nabla F(a_n)$ with γ a positive real number. In artificial intelligence it is named the "learning rate" as it impacts the convergence speed. Hopefully, we should have $F(a_n) \geq F(a_{n+1})$
4. Compute the gradient of F in a_{n+1}

5. Repeat step 3 and 4 until we reach convergence, which can be for instance defined by $||a_{n+1} - a_n|| \leq \varepsilon$ where ε is a small value ($1e-3, 1e-4 \ldots$)

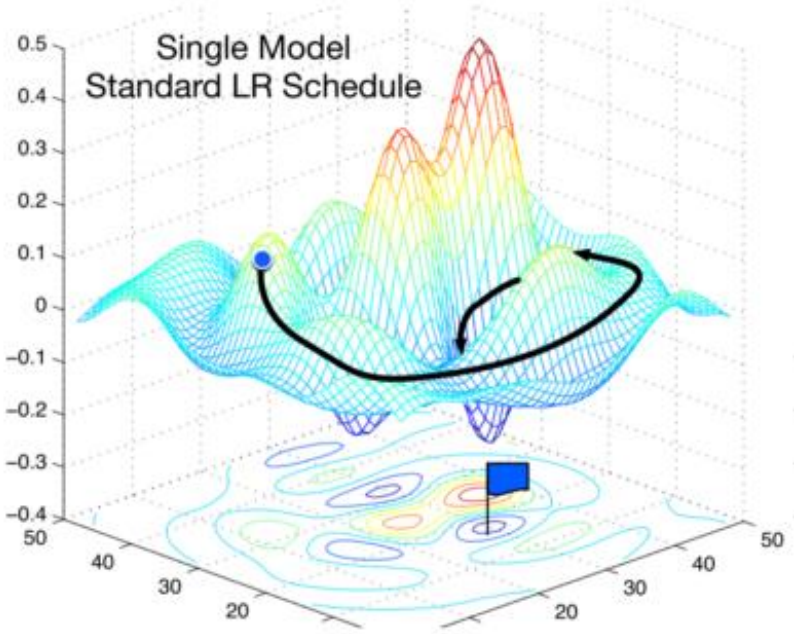


Figure 11 – Gradient descent on a 2 parameters function. The negative gradient points towards the maximum decrease. This hopefully leads to the global minimum

2.4.2.2 Limitations

This works well with convex functions (if any two points of it are taken and a line is drawn to link them, then all points between them are above or under this line), because any local minimum is also a global minimum. But in many different cases, this is not the case: there are many local minima in which the gradient gets trapped [35, 36]. A way to avoid this is to set a high learning rate. But having a high learning rate leads to other issues. Thus, in that case, one solution is to randomly choose a set of different points and start a gradient descent from each one of those sets to hopefully reach the best result.

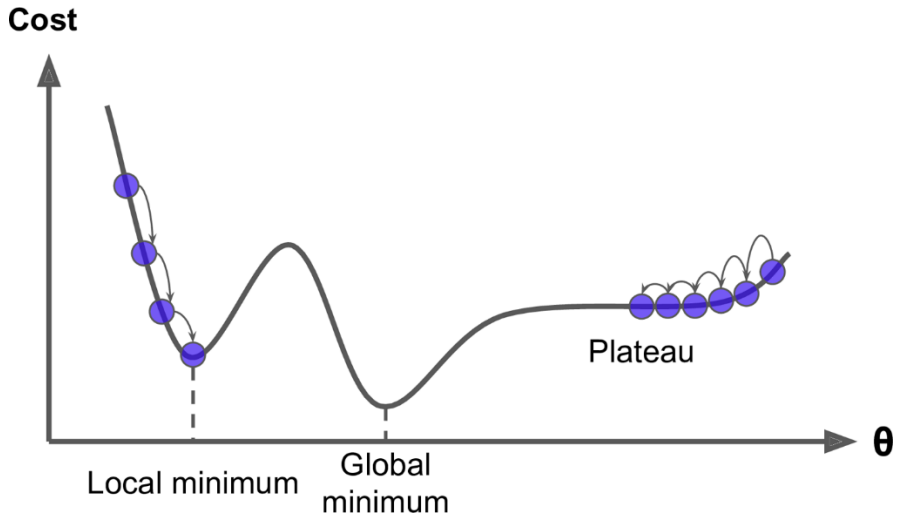


Figure 12 – This illustrates how the gradient can get trapped. On the left, the algorithm reaches the local minimum. The gradient has no way of getting out of this minimum, as soon it goes back on the slope, the gradient will point again towards the local minimum. On the right, the algorithm gets to a plateau. That means the variation from one point to another gets very small, possibly inducing the program into error by making it believe the minimum is reached.

2.4.3 Second order optimization algorithms

Second-order optimization algorithms rely on the use of the second derivative (which physically represents the acceleration of the signal) and the Hessian array. Its main drawbacks are that it is costly in terms of calculation (the inversion of an n * n Hessian matrix has a complexity of $O(n^3)$, from section 8.6 of [37]), and that it is attracted by saddle points [38] (usually pretty common in optimization problem [38]). The latter is a point where the gradient is null, but which is not a local/global minimum. On the other hand, second order optimization algorithm has the advantage of quadratic termination: it gets the minimum of a function in a finite number of steps. This is not the case for the gradient, which in the worst case diverges if the sampling rate is not chosen appropriately.

Method (from [39, 40]): let's assume we have a multivariable function F, which is defined and twice-differentiable for any real value noted X of those parameters

1. Given X^0, set k = 0
2. Set $d^k = -H(x^k)^{-1}\nabla f(x^k)$. If $d^k < \varepsilon$, where ε is a small value, then stop
3. Choose a step size $\propto^k = 1$
4. Set $x^{k+1} = x^k + \propto^k d^k$, and start again to step 2

For more details and justifications, refer to [40]

Based on the previous discussion, we concluded that:

- The frequency approach seems here more appropriate to solve our problem than the time domain approach. Indeed, the latter is not very appropriate for determining the nature of the defect and getting back the bearing parameters. Therefore, all our analysis is based on Fourier transform
- Concerning the optimization methods: the plan was initially to use a mix of zero and first orders. Indeed, as explained later, the algorithms we rely on use a set of parameters that may impact the performances. Nonetheless, after exploratory work, it appears that the space solution is full of local minimums blocking gradient descent. Using second order optimization does not solve the issue either because of the multiple saddle points. Hence, random search is used

CHAPTER 3: EXPERIMENTAL BACKGROUND

3.1 Context and assumptions

The defect frequencies are known thanks to the theoretical equations. Nonetheless, there is always a difference with what they are in practice. There are several reasons for this:

- The formulas are computed assuming no rolling element slippage. But this in fact is not the case. Slippage is usually around 1-2% [4]
- The RPM value is sometimes partially known. For instance, this can happen in the paper industry. The paper sheet is going through a group of rollers. Since the latter are supposed to turn approximately at the same speed, in some factories it has been decided that only one tachymeter would be used to measure the rolling speed for that group of rollers. But due to multiple different phenomena, there may be a difference over 20 - 40% between the measurement and the real RPM of some rollers.
- Usually, the other parameters (rolling element diameter, pitch diameter …) are known, but this is not always the case.

Sometimes the difference between theory and practice has a negative impact on defect detection: the latter is often realized by software which assumes that all the parameters given are the real ones. Therefore, software algorithms only look in a restricted area around the theoretical values and miss the signature, which may be out of the search zone. Thus, a flag is not triggered, and the consequences can be costly: the risk is to

discover the issue at an advanced stage of wearing (worse, the bearing fails), forcing the maintenance team to respond in emergency. Right now, solving the fact information is lacking is costly (adding other sensors, hiring personnel to monitor visually the spectra…) or difficult: modification of the software (one can add this is not often possible).

Therefore, the goal of this thesis is to present a solution able to detect a BPFO /BPFI /BSF /FTF defect assuming we have an approximate knowledge of the 5 defect frequencies parameters. The latter vary in the following range:

- RPM value: ranging from -40% to 40% of the measurement. The difference between theory/practice is supposed to be due to phenomena having a limited impact on the overall speed (slippage, inaccurate sensor…) not to phenomena that modify completely the final value
- Number of rolling elements: - 2 elements to + 2 elements. This is difficult to imagine a case where the variation is bigger. Larger variations would imply the use of a very different bearing type, hence, there is little chance this bearing would fit into the constrained system (shaft diameter and outer race exterior diameter are imposed)
- Contact angle: -5° to +5° and pitch diameter, rolling element diameter: - 5% to +5%. This assumption seems quite reasonable. Bigger differences with the theoretical value make no sense: either we have no information on the bearing (this situation is not studied in this thesis), or we know what kind of bearing can fit into the machine, and thus we should have a relatively reliable approximation.

3.2 Reduction of parameters

From the defect formula, one can see that, from the assumption we made in 3.1, some parameters have a neglectable / small impact on the theoretical BPFO/BPFI/FTF/BSF value.

3.2.1 Contact angle

From equation (13) we can prove contact angle has a minor impact on final BPFI value (similar demonstration can be made for BPFO, BSF, FTF). First, as indicated in [41, 42], we will assume the contact angle is between 10 and 45°, as those are the most common values. Let's note β the variation in degrees of contact angle.

$$\mathbf{cos(\alpha \pm \beta) = cos(\alpha) cos(\beta) \pm sin(\alpha) sin(\beta)} \quad \textbf{(26)}$$

$$\mathbf{-5° \leq \beta \leq +5° => cos(\beta) \geq 0.996} \quad \textbf{(27)}$$

Therefore, we simplify the equation by setting $\cos(\beta) = 1$. From (28) and (29):

$$\mathbf{bpfi = Z * \frac{f}{2} * \left(1 + \frac{D}{d_m} cos(\alpha) \pm \frac{D}{d_m} sin(\alpha) sin(\beta)\right)} \quad \textbf{(28)}$$

Let's prove the ratio $\left|\frac{\frac{D}{d_m}\sin(\alpha)\sin(\beta)}{1+\frac{D}{d_m}\cos(\alpha)}\right|$ is close to 0, implying that the contact angle variation can be neglected. Plotting this ratio, we get the maximum for $\alpha = \pm 45°, \frac{D}{d_m} = 0.5,\ \beta = \pm 5°$, with the ratio equal to 0.022. Therefore, the uncertainty of the contact angle impacts by a maximum of 2.2% the final value of BPFI. Knowing the variation of the number of rolling elements / RPM can impact the BPFI value by more than ten times this factor, we choose to neglect the contact angle uncertainty.

3.2.2 Rolling element diameter

Let's assume the worst case: to have maximum impact on BPFI value, the term $|\frac{D}{d_m}\cos(\alpha)|$ must be maximized. We will therefore assume $\alpha = 0$.

From (13), this gives us $1 + \frac{D(1 \pm \beta)}{d_m}$ with $\beta \leq \ 0.05$.

Besides, a reasonable assumption is to say that the diameter of the rolling element is at most the half of the pitch diameter (because if not, then it implies the shaft diameter plus the thickness of the inner race times 2 are smaller than the rolling element's diameter itself):

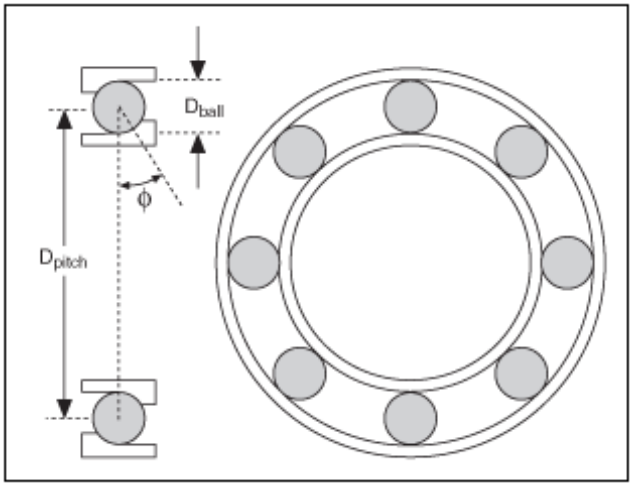


Figure 13 – D_{pitch}, in most of the cases should be twice bigger than D_{ball}

Thus,

$$1 + \frac{D(1 \pm \beta)}{d_m} = 1 + \frac{D}{d_m} \pm \beta * \frac{D}{d_m}$$

Let's prove that the term $\beta * \frac{D}{d_m}$ is not going to have an important impact on the final result. This implies showing that $\frac{\beta * \frac{D}{d_m}}{1+\frac{D}{d_m}}$ is negligible. As previously said, $\frac{D}{d_m}$ varies from 0 to 0.5. Finding the maximum is the same as finding the maximum of $\beta * \frac{x}{1+x}$ which is located for $x = 0.5$ (proof done by derivation the function on [0, 0.5] segment). Henceforth, using our assumptions, we get:

$$\frac{\beta * \frac{D}{d_m}}{1 + \frac{D}{d_m}} \leq \frac{0.05 * \frac{1}{2}}{1 + \frac{1}{2}} = 1.7\%$$

This implies that, in case of rolling element diameter variation, the defect frequencies are going to be impacted by not more than 1.7%. The assumption of ignoring the rolling element diameter variation appears relevant, following the same reasoning as for the contact angle.

3.2.3 Pitch diameter

From the equations, one can derive that the pitch diameter variations are going to impact the final value of the defect frequency by a higher percentage than contact angle / rolling element diameter. Nonetheless, using our assumptions, this should be relatively smaller than the RPM or number of rolling elements impact. Therefore, its impact will be studied, but in a second time, once we know we are able to find defected spectra when the RPM and the rolling elements values are uncertain.

3.3 Methodology

3.3.1 Data

Data acquisition and manipulation is a critical step in our process because the quality of our data often impacts the results. We will work here with artificial data / generated data. Many criticisms have been raised about such practices: not realistic, too easy, etc. However, artificial data also has the advantage of being reliable in the sense that one controls all the parameters of the signal. Thus, the result can be compared to a reliable reference, which is not the case with signals extracted from physical systems. Besides, it is much more flexible. It is possible to test multiple different configurations in a limited time, and to work with data that we could not get from a physical system because some of its parameters can't be changed. In the case of a bearing, it is for instance, usually not possible to change the number of rolling elements. Making the latter vary implies buying a set of different elements. Obviously, this takes time, and money and it is not always possible to get the exact same system with only one parameter changing. With generated data, it is only a matter of variables that can be easily modified.

3 different type of data are created:

- Healthy data: 180 white noise signals
- BPFO data: 30 signals. There are 10 different stages of wear (so there will be 3 spectra per wear stage), to stick as much as possible with reality
- BPFI data: 30 signals. As for BPFO, the signals are also divided in 10 different groups

We have included a significant amount of healthy data to make sure our algorithm is not too sensitive

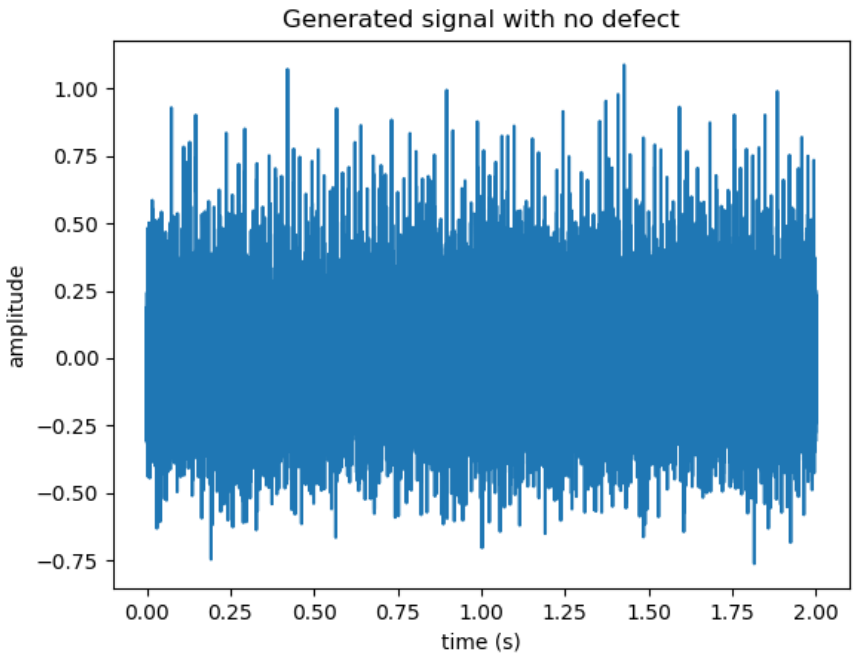


Figure 14 – Artificial data generated with no defects

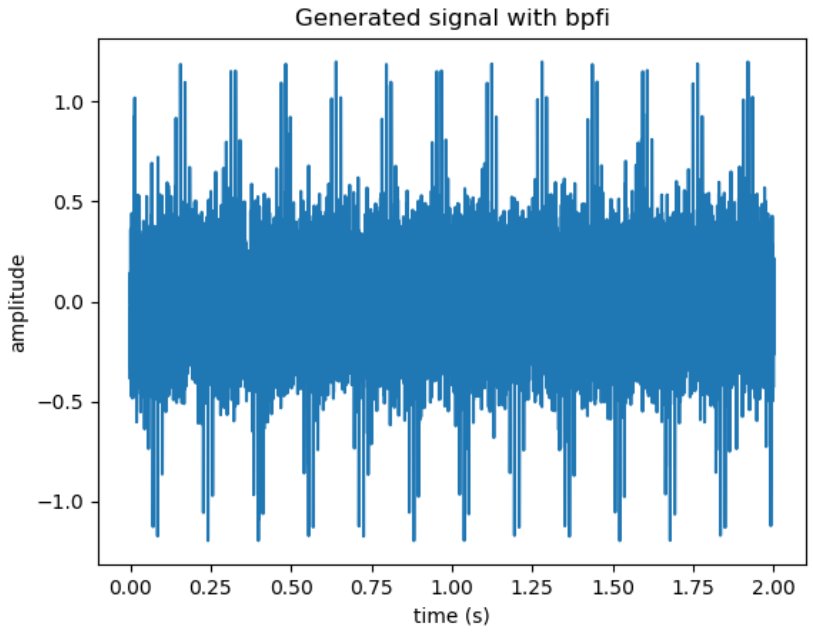


Figure 15 – Simulation of a BPFI defect

3.3.2 Tools

- Python v3.6: it is used for all the data creation, data processing and analysis. The main libraries used for this purpose are numpy, scipy and pandas.
- Excel: used for data visualisation during the experiment and research part.

3.3.3 Evaluation of results

One of the challenges in some scientific experiments is evaluating the quality of the results obtained, especially when no similar work has been done (no comparison possible). In our context, the difficulty comes from the fact we do not have a single unique parameter to evaluate, but two:

- Type 1 error: more known under the term “false positives”. Here type 1 error evaluates the number of cases where the spectrum is flagged as having a defect though it is healthy.
- Type 2 error: also known under the term of “false negative”. It evaluates the number of spectra that were not flagged as having a defect though in reality they do have one.

Ideally, both errors should be very close from zero. The issue is that type 1 and type 2 errors are inversely proportional: if one goes down the other goes up. This can be easily understood: to avoid false positives, one applies conditions that have to be respected by the studied spectrum in order to be flagged as a defect. As a consequence, even some of the spectra representing signals of defected elements are missed. This is especially true if the default is recent (the spectral signature is not very pronounced). On the other hand, reducing type 2 error implies loosening the constraints, causing a certain number of spectra looking like defects to be incorrectly flagged.

Therefore, a trade-off must be found. Additional context is then needed to find a suitable ratio. Indeed, according to the case, one gives more importance to type 1 or type 2 values. In the industrial world, low values for type 2 error and medium values for type 1 error are likely to be preferred: an alarm triggered for a false positive has few consequences, a fault missed or detected too late is much more of a problem. Therefore, we will emphasize false negative errors and try to minimize these while keeping an acceptable amount of false positives.

3.3.4 Different steps of experimentation

The experiments are organised as follows:

1. The first experiment is done assuming the only uncertainty is on the RPM. Therefore, the goal is to analyse the signal, determine if there is a defect, and if so, compute back the most likely RPM. The same experiment could also be done to compute back the number of rolling elements. It would be strictly identical; therefore, we chose to focus only on RPM (anyway, in most cases, the uncertainty should be on the latter).
2. The second experiment is done assuming uncertainty on RPM and number of rolling elements. Then, the same process is followed as in 1), with the additional step of giving the most suitable number of rolling elements.
3. The third experiment is done assuming uncertainty on RPM, number of rolling elements and pitch diameter. Variations in other geometrical parameters are ignored, since we demonstrated their relatively low impact on the final value of defect frequency. Again, the same process is used as in 1) and 2). The same obstacles as in 2 (multiple working combinations) have to be faced.

For every experiment, the accuracy of the analysis is evaluated (type 1 and type 2 error), and then the precision of the parameters' estimation is given.

3.4 Explanation of the defect detection process

The defect detection process is the following:

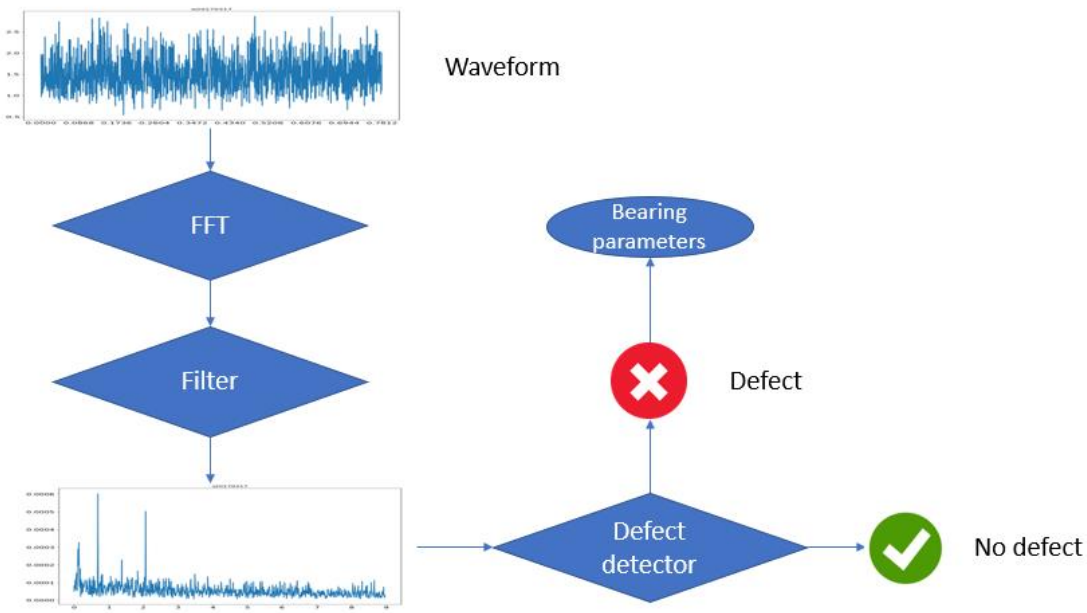


Figure 16 – Defect detection process

So far, FFT transform and filter have been explained. Let's discuss about the defect detector and the recovery of bearing parameters.

First, the program takes as an input the Fourier transform of the signal. Second, the frequency axis is divided by the theoretical defect frequency. If a fault is present, then, the largest amplitudes should be located around the integer values of x-axis. Therefore, the algorithm gathers a list of peaks around those areas and divides them into groups corresponding to an integer value, i.e. 1 * defect frequency, 2 * defect frequency (= first harmonic) etc. If a peak is approximately as close as two multiples of the fault frequency, then it can belong to 2 groups simultaneously:

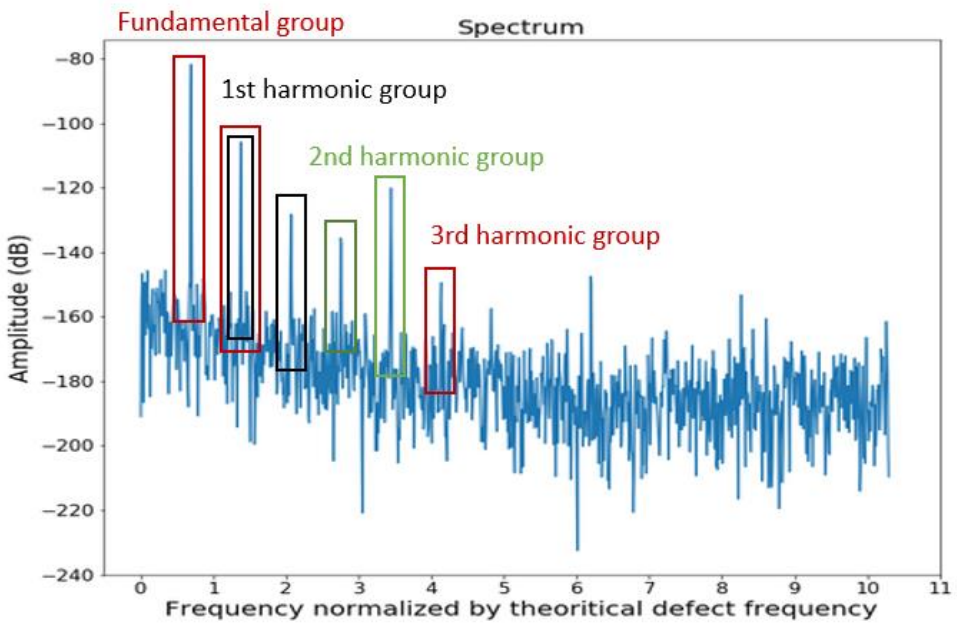


Figure 17 – Gathering and grouping peaks in integer groups

Then, different sets of peaks are built: 1 peak represents the fundamental, 1 represents the first harmonics, another the second…: We assume in turn that the fundamental of each set is the defect frequency and divide the x-axis of the raw spectrum

by this value. The set error is computed by summing the set peaks' distances to their respective integers:

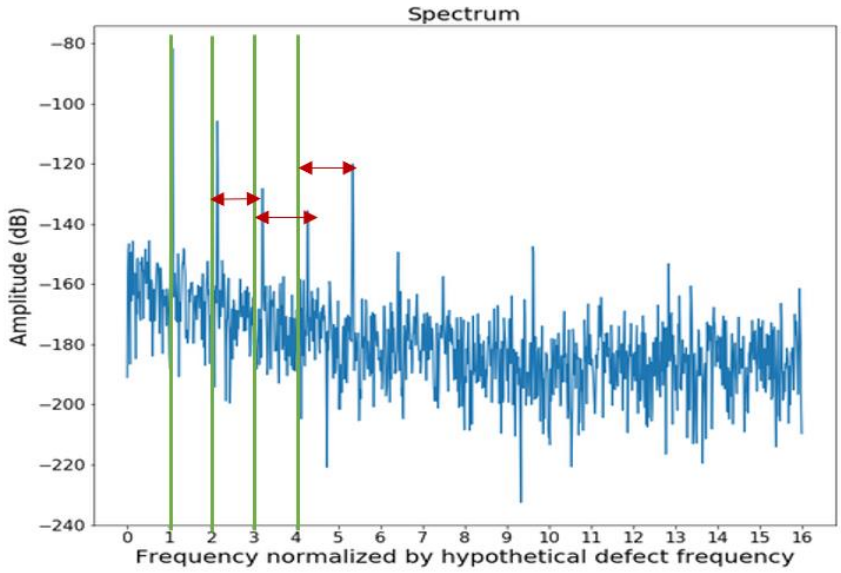


Figure 18 – Normalization by fundamental of set 1. The distance integer-peak is high, there is a small likelihood this set represents the defect signature

The set with the best score is taken, and if its error is below a threshold (defined by the user), then the bearing is flagged. Note that other criteria are tested to check if the set really represents a defect signature (size of peaks compared to the spectrum mean, difference of size between fundamental and harmonics etc.). Depending on defect type, other features are implemented.

If no error is detected, then the process stops. On the contrary, if a flag is raised, then the uncertain parameters are computed using the formulas (13), (14), (15), (16). An n-space solution is explored, and the best parameter combinations are given.

CHAPTER 4: EXPERIMENT AND ANALYSIS

4.1 Experiment 1: RPM as the only approximate parameter

4.1.1 Results

We have the following results:

Table 1 – Confusion matrix

	Labelled as BPFO	Labelled as BPFI	Labelled as no defect
BPFO spectrum	87%	0%	13%
BPFI spectrum	17%	67%	17%
No-defect spectrum	2.4%	1.9%	95.7%

Once the defects are identified, we compute the RPM back (supposed to be equal to 360 rotations per minute). Using the spectra that were labelled as having a ball pass outer defect one gets:

Table 2 – Estimation of the exact RPM value (working with BPFO flagged spectra)

Type of spectrum on which the calculation is made	RPM computed back	Difference with the real value (360 rotations/min)
BPFO spectrum	357	1%
BPFO spectrum	363	1%
BPFI spectrum	460	28%
No-defect spectrum	631	75%
No-defect spectrum	652	81%
No-defect spectrum	686	91%

Concerning the BPFI labelled spectra:

Table 3 – Estimation of the exact RPM value (working with BPFI flagged spectra)

Type of spectrum on which the calculation is made	RPM computed back	Difference with the real value (360 rotation/min)
BPFI spectrum	363	1%
No-defect spectrum	602	67%
No-defect spectrum	619	72%
No-defect spectrum	629	75%
No-defect spectrum	725	101%

4.1.2 Analysis

Different aspects have to be mentioned:

- Concerning defect detection part, the type 2 error of the BPFO algorithm is lower than that of the BPFI one (87% successful detection vs 67%). On the other hand, type 1 error is higher (17% of the BPFI and 2.4% of the no-defect spectra labelled as BPFO versus 0% of the BPFO and 1.9% of the no-defect labelled as BPFI). The difference can be attributed to two factors. The first one

lies in the compromise that has to made with type 1 and type 2 error. Here, we chose some parameters that tend to favor a low rate of BPFO spectra missed and a low rate of misclassified BPFI data. The second reason explaining the difference is that by nature the BPFI signature has more constraints than its counterpart: it requires the spectrum to present clear sidebands. In some cases, the latter tend to be hidden by noise, making it harder for the process to flag. The good side of this is the lower rate of false positives (likelihood of a random noise presenting clear sidebands and peaks is relatively lower).

Computing the true value of the RPM back is successful here. The false positive cases do not impact much the final result. Indeed if a spectrum is flagged even though it comes from a healthy bearing, one gets as an output an RPM value which is far (>67%) from the predicted value. Besides, it varies from one signal to another significantly. Hence, it is relatively straightforward for the algorithm to discard those signals:

- Table 2 and Table 3 clearly emphasize that if there is a bearing defect, then the RPM value found by the algorithm does not differ much (less than 2%) from one signal to another. Therefore, if the estimated rotation speed is not constant, then it can be guessed that the waveform is a false positive.

To conclude this first part, one can say getting an accurate value of the RPM when a defect is detected is feasible. Additionally, we have a satisfactory rate of defect detection.

4.2 Experiment 2 – RPM and number of rolling elements' true values unknown

Here, we assume we have a partial knowledge both on the RPM (±40%) and the number of rolling elements (±2 elts). Each spectrum is tested to see whether a defect is present or not for each possible number of rollers.

4.2.1 Results

One gets the type 1 and type 2 error for each number of rollers. In red, the cases where the detection rate differs from experiment 1:

Table 4 – Confusion matrix

	Labelled as BPFO					Labelled as BPFI					Labelled as no defect				
BPFO spectrum	18	19	20	21	22	18	19	20	21	22	18	19	20	21	22
	87%	87%	87%	87%	83%	0%	0%	0%	0%	0%	13%	13%	13%	13%	17%
BPFI spectrum	18	19	20	21	22	18	19	20	21	22	18	19	20	21	22
	17%	17%	17%	17%	17%	67%	67%	67%	63%	63%	17%	17%	17%	21%	21%
No-defect spectrum	18	19	20	21	22	18	19	20	21	22	18	19	20	21	22
	2.4%	2.4%	2.4%	2.4%	2.4%	1.9%	1.9%	1.9%	1.9%	1.9%	95.7%	95.7%	95.7%	95.7%	95.7%

If a defect is found, then, what should be the true RPM is computed back. The difference with experiment 1 is that we do not get a single solution:

Table 5 – Different working pairs {number rolling elements, RPM} for BPFO flagged spectra

Number of rolling elements	RPM found	Difference with theoretical RPM (300)
18	335	12%
19	317	6%
20	301	0.3%
21	287	4%
22	274	9%

Then, for the BPFI defect detection, we get similar results:

Table 6 – Different working pairs {number rolling elements, RPM} for BPFI flagged spectra

Number of rolling elements	RPM found	Difference with theoretical RPM (300)
18	332	11%
19	314	5%
20	299	0.3%
21	284	5%
22	271	10%

The main difference between the outer and inner race defect signatures are the sidebands. The latter are separated from one another by a distance of 1 RPM frequency. Thus, in theory, one should be able to retrieve back the rotation speed by simply looking at the Fourier transform. Therefore, the algorithm looks for the RPM using the sidebands:

Table 7 – RPM found using the sidebands

Number of rolling elements	18	19	20	21	22
RPM found	362	362	362	362	362
Variation with true value (300)	21%	21%	21%	21%	21%

4.2.2 Analysis

We now are in a more complex case: we have 2 unknowns and 1 equation. If no additional information is provided, it is impossible to get back the right bearing configuration. Nonetheless, as Table 4 underlines, the diagnostic remains very similar. It means the variation of the number of rolling elements is too small to have a meaningful impact on defect detection rate. The spectra for which classification has changed between experiments 1 and 2 are edge cases: the input RPM is around 40% larger than what it is in reality. Therefore, if the number of rolling element increases (+1 or +2), it raises by a few percent the theoretical BPFO/BPFI value. The peaks corresponding to the latter get out of the search zone making the program asserting the bearing is healthy.

Some configurations could be eliminated by counting the number of flagged spectra once we know with high confidence that the bearing is unhealthy and getting rid of configurations with lower flagged spectra rate.

Table 4 shows that this approach may not be very effective though: at maximum, only 2 configurations can be eliminated, and the difference is only made on 1 signal.

One can remark the main difference between the outer and inner race defect signatures are the sidebands. The latter are separated from one another by a distance of 1 RPM frequency. Thus, in theory, one should be able to retrieve back the rotation speed. In practice, this is not really the case, at least for our signals: the RPM found for each case differs significantly from the real speed (21%). The difference might be explained by the low number of sidebands (at most 1), preventing averaging (which reduces the error). The detection of the sideband itself may also be imperfect: the wrong peaks may have been chosen as sidebands, therefore reducing the quality of the final output. Ultimately, the resolution is not optimal, increasing the difference with what is expected by a few percent.

4.3 Experiment 3: RPM, number of rolling elements and pitch diameter true values unknown

This time, we add the fact we have an incomplete knowledge of the pitch diameter ($\pm$5%). Let's check if we are still able to get some equivalent results.

4.3.1 Results

Table 8 – Different working pairs (number rolling elements, RPM) for BPFO flagged spectra

Number of rolling elements	Pitch diameter (mm)	RPM found	Difference with value found in experiment 2
18	335	338	0.9%
	340	337	0.6%
	345	336	0.3%
	350	336	0.3%
	355	335	0%
	360	334	0.3%
	365	334	0.3%
	370	333	0.5%
19	335	320	0.9%
	340	319	0.6%
	345	319	0.3%
	350	318	0.3%
	355	317	0%
	360	317	0%
	365	316	0.3%
	370	316	0.5%

Table 8 – Continued

20	335	304	1%
	340	303	0.7%
	345	303	0.7%
	350	302	0.3%
	355	302	0.3%
	360	301	0%
	365	300	0.3%
	370	300	0.3%
21	335	290	1%
	340	289	0.7%
	345	288	0.3%
	350	288	0.3%
	355	287	0%
	360	287	0%
	365	286	0.3%
	370	286	0.3%
22	335	276	0.7%
	340	275	0.4%
	345	275	0.4%
	350	274	0%
	355	274	0%
	360	273	0.4%
	365	273	0.4%
	370	272	0.7%

Same is done with BPFI detection:

Table 9 – Different working pairs (number rolling elements, RPM) for BPFI flagged spectra

Number of rolling elements	Pitch diameter (mm)	RPM found	Difference with value found in experiment 2
18	335	338	0.9%
	340	337	0.6%
	345	336	0.3%
	350	336	0.3%
	355	335	0%
	360	334	0.3%
	365	334	0.3%
	370	333	0.5%
19	335	320	0.9%
	340	319	0.6%
	345	319	0.3%
	350	318	0.3%
	355	317	0%
	360	317	0%
	365	316	0.3%
	370	316	0.5%

Table 9 – Continued

20	335	304	1%
	340	303	0.7%
	345	303	0.7%
	350	302	0.3%
	355	302	0.3%
	360	301	0%
	365	300	0.3%
	370	300	0.3%
21	335	290	1%
	340	289	0.7%
	345	288	0.3%
	350	288	0.3%
	355	287	0%
	360	287	0%
	365	286	0.3%
	370	286	0.3%
22	335	276	0.7%
	340	275	0.4%
	345	275	0.4%
	350	274	0%
	355	274	0%
	360	273	0.4%
	365	273	0.4%
	370	272	0.7%

4.3.2 Analysis

The results we have are almost identical between experiments 2 and 3. This can seem surprising, but when one has a look at Table 8 and Table 9 the reason is clear. Even when extreme cases are taken (pitch diameter is $\pm 5\%$ different from reality) the impact on the final RPM does not exceed 1%. This low impact on final RPM implies the real defect frequencies are not very different from what we had in experiment 2. Thus we get similar results.

This finding emphasizes that geometrical bearing parameters do not play the most important role concerning the defect frequencies (provided a variation in a range of $\pm$ 5%). Consequently, we again face the issue of finding the true set {RPM, number of rollers} that corresponds to the physical reality. All the 5 sets provide a credible result and finding the right one with high confidence rate is not possible in our case. In a way this is not too much of a big deal: in most cases flagging the fault is the only thing that matters.

CHAPTER 5: CONCLUSION

5.1 Contribution of this thesis

The contributions of this thesis are:

- Proposing a procedure for fault detection when one has an imperfect knowledge of the system and exploring solutions to get some parameters back. It seems that if either RPM or number of rolling elements is missing, finding its real value back is not of big deal. The approximation is trusted with a high confidence. However, if more than one parameter is uncertain, it gets much more difficult to get back to the true configuration. Concerning the detection rate, it has an impact, slightly reducing the number of successful identifications. But the overall results remain quite satisfactory.
- Proposing an automated algorithm of defect detection and studying their performances. In particular, it has been found that the BPFO algorithm performs better in terms of type 2 error, but it has a higher rate of false positives.

5.2 Improvements and limitations

The work done here implies a certain knowledge of the bearing. Some arbitrary limits have been set (the $\pm$ 5% uncertainty on geometrical parameters, the $\pm$ 2 rolling elements variation). It could be interesting to explore cases where one does not know the bearing configuration. In case of success this would provide a very flexible tool for industry, as it is sometimes difficult to get the data about the bearing itself. Besides, some companies face

the issue of having a large variety of the latter, implying a long, fastidious and subject to error task of setting the different variables for each element.

The algorithms used rely on a statistical approach in order to detect the peaks. This part is crucial for successful identifications. If it fails, type 1 error can reach unacceptable levels. The statistic method may appear strong and reliable, but in practice this is not the case. Many phenomena can make it very difficult to identify the right peaks: these phenomena can include the ski slope issue, bursts in high frequencies, peaks appearing due to other components' vibration etc. The latter provokes a raise of global mean and standard deviation. In consequence, some peaks go undetected because they are below a certain threshold (ex: mean + n * standard deviation, with n a positive real number). Using a local mean/standard deviation does not solve all the problems, as it requires the user to define arbitrarily a certain range on which to compute it. In the end, to make a robust detection system, one has to use brute-force by taking into account all possible cases. Another interesting approach would be to use artificial intelligence to detect peaks / to detect faults, in particular convolutional neural networks (CNN). The latter performs well in image recognition and could be used as an additional way to strengthen the detection.

The detection is only based on the Fourier transform. However, other tools can be used: the control charts (though relatively imprecise on the defect nature) and the waveform. Identifying the latter a fault represents a more challenging task, the use of deep learning would probably be required. But from discussion with experienced engineers working in this sector, the temporal signal is definitely something they have a look at when doing a rolling element diagnostic.

In this thesis we have not measured the sensitivity of the algorithm, i.e. the evolution of performances according to the parameters' uncertainty. In other words, if we assume for instance that the number of rolling elements could this time vary from ± 5 or ± 10 rolling elements, would our type 1 and type 2 error be modified? If yes, which parameter's uncertainty range would have the largest impact on the results? If no, then up to which uncertainty range can we go to without having any impact?

Finally, only generated data has been used for the experiments. Using data coming from real systems with reliable knowledge of all the different parameters would be very interesting, bringing new challenges to face.

5.3 Conclusion

This thesis has presented an automated approach to detect bearing faults. Indeed, as said previously, physical models have been developed, but some blocking points still exist. Due to inaccurate parameter values, the automated systems are not able to realize a reliable diagnostic. As a matter of consequence, we developed solutions to overcome this issue. We have shown that we are able to easily combine defect detection and RPM recovery when the latter is the only parameter on which we have uncertainty. When the number of parameters on which we have a partial knowledge increases, we are not able anymore to provide with a high certainty what are the real values. But we still can make correct predictions, which in the end is probably the thing that matters most. Finally, we discussed about the limitations and improvements that could be brought to the existing system.

APPENDIX A. DESCRIPTION OF DEFAULT SUBHEADING SCHEME

A. 1. Data generation codes

```
# Data generation
# imports
import sys
sys.path.append("../../Recherche/Développement/Spindle_Bearing_Analysis/vibration_analytics")
from rollerbearing import RollerBearing
from vibmachine import VibMachine
import matplotlib.pyplot as plt
import numpy as np
import traceback

# signal parameter
samplingRate = 4096
nbPts = samplingRate * 5

# rolling element parameters:

rpm = 360
numElements = 20
pitchDiam = 50
ballDiam = 10
contactAngle = 15

bearingConfig = {"ballDiam" : ballDiam,
   "pitchDiam" : pitchDiam,
   "numElements" : numElements,
   "contactAngle" : contactAngle}

rollBear = RollerBearing(samplingRate, rpm, **bearingConfig)

# get default defect frequencies
rollBear.getDefectFreq(rpm)
```

```
# generation of fundamental and harmonics
def genDefectFreq(ampFonda, percVarAmp = 0.3, ratioValFondastd = 0.1, nbHarmo = 5):
    amps = np.zeros((1 + nbHarmo))
    amps[0] = ampFonda

    for k in range(1, nbHarmo + 1) :
        # value around which our amplitudes are generated
        loc = ampFonda * (1 - percVarAmp)
        amps[k] =  np.random.normal(loc, scale = ratioValFondastd * ampFonda)

    return(amps)


# création d'une instance vibmachine
myVib = VibMachine(samplingRate, rpm)


def genSign(samplingRate, nbPts, myVib) :
    # white noise generation
    sign = np.random.normal(0, scale = 0.2, size = nbPts)

    return(sign)
```

```
def genBpfo(sign, samplingRate, freqBpfo, dureeImpact = 10, ampliImpact = 0.5):
    # fix the seed to always get same results
    np.random.seed(2)

    # sign = temporal signal to transform
    n = len(sign)
    harmo = 1
    for k in range(1, n):
        # check we are in a zone of impact
        if np.abs(k * 1 / samplingRate  - harmo * 1 / freqBpfo) < 1 / samplingRate:

            # modification in that zone an amplitude
            index = np.random.randint(-3, 4)
            for j in range(dureeImpact) :
                if k + index + j < n :
                    sign[ k + index + j] = \
                    (ampliImpact + np.random.normal(0, scale = 0.3)) * 10 ** (- j / dureeImpact / 5)
            # not really an harmo, represents the number of times where the defect was hit by a rolling
            # element
            harmo += 1
    # gets back to waveform
    return(sign)
```

```
############### Example : generation of bpfo data ####################

freq = rollBear.ballPassOuterFreq

sign = genSign(samplingRate, nbPts, myVib)

bpfoData = genBpfo(sign, samplingRate, freq, ampliImpact = 0.5)


freqs, amps = myVib.generateFFT(bpfoData)

freqs1, amps1, timeAxis, enveloppe = myVib.autoCorrAndRedressWaveform(samplingRate, sign, 4 *
freq, orderFilt=1)


plt.plot(bpfoData[ : 1000])

plt.title("waveform with of bpfo bearing")

plt.show()


plt.plot(freqs[ 20 : 500] /freq, amps[ 20 : 500])

plt.title("Spectrum of bpfo bearing")

plt.show()


plt.plot(freqs[ 20 : 500] /freq, 20*np.log(amps[ 20 : 500]))

plt.title("Spectrum of bpfo bearing")

plt.show()


plt.plot(freqs1[ 20 : 500] / freq, amps1[ 20 : 500])

plt.title("Filtered spectrum of bpfo bearing")

plt.show()


plt.plot(freqs1[ 20 : 500] / freq, 20 * np.log(amps1[ 20 : 500]))

plt.title("Filtered spectrum of bpfo bearing")

plt.show()
```

```
########################## bpfi ###################################
def genBpfi(sign, samplingRate, freqBpfi, rpm, dureeImpact = 10, ampliImpact = 0.5):
   #np.random.seed(2)
   # sign = signal temporel à transformer
   n = len(sign)

   # not really an harmonic, just displaying the number of times we passed by the defect
   harmo = 1
   for k in range(1, n):

      if np.abs(k * 1 / samplingRate  - harmo * 1 / freqBpfi) < 1 / samplingRate:
         index = np.random.randint(-1, 2)
         # modulation of signal amplitude
         # amp = A * cos(2 * pi * t  / T)
         # T * omega = 2 * pi and omega = rpm /60 * 2 * pi
         T =  60 / rpm
         ampli = ampliImpact * np.cos(k * 1 / samplingRate * 2 * np.pi / T)

         for j in range(dureeImpact) :
            if k + index + j < n :
               sign[ k + index + j] = \
               ampli  * 10 ** (- j ** 2/ dureeImpact ** 2)
          #"""

         harmo += 1
   # return waveform
   return(sign)
```

```
############### Example : generation of bpfi data ####################


for k in range(5) :

    bpfiData = genBpfi(sign, samplingRate, freq, rpm, dureeImpact = 10, ampliImpact = 2)

    freqs, amps = myVib.generateFFT(bpfiData)

    freqs1, amps1, timeAxis, enveloppe = myVib.autoCorrAndRedressWaveform(samplingRate, sign, 4
* freq, orderFilt=1)

    #"""

    plt.plot(bpfiData[ : 3000])

    plt.title("waveform with of bpfi bearing")

    plt.show()


    plt.plot(freqs[ 20 : 500] /freq, amps[ 20 : 500])

    plt.title("Spectrum of bpfi bearing")

    plt.show()


    plt.plot(freqs1[ 20 : 500] / freq, amps1[ 20 : 500])

    plt.title("Filtered spectrum of bpfi bearing")

    plt.show()


    plt.plot(freqs1[ 20 : 500] / freq, 20 * np.log(amps1[ 20 : 500]))

    plt.title("Filtered spectrum of bpfi bearing")

    plt.show()


    plt.plot(freqs[ 20 : 500] /freq, 20*np.log(amps[ 20 : 500]))

    plt.title("Spectrum of bpfi bearing")

    plt.show()
```

A. 2. Experiments

```
##################EXPERIMENT 1 : we have a partial knowledge of the rpm###################################

from ast import literal_eval

# now we have the data, we can start the experimentation part

df = pd.read_pickle("data.pkl")

n = len(df)


#np.random.seed(1)


# type 1 = false positive , type2 = missed a bearing with a defect

score = {"type1Bpfo" : 0, "type2Bpfo" : 0, "type1Bpfi" : 0, "type2Bpfi" : 0, \

      "totalNoDefect" : len(df[df["defect"] == "no"]), \

      "totalBpfo" : len(df[df["type"] == "bpfo"]), "totalBpfi" : len(df[df["type"] == "bpfi"])}


estimationRpmDic = {"bpfo" : [], "bpfi" : []}



bearingConfig = {"ballDiam" : ballDiam,

   "pitchDiam" : pitchDiam,

   "numElements" : numElements,

   "contactAngle" : contactAngle}


for k in range(n) :


   # generation of fft

   waveform = np.array(df['data'][k])


   freqs, amps = myVib.generateFFT(waveform)
```

```
    # generation of random rpm
    estimateRpm = rpm * (np.random.randint(-40, 41) / 100 + 1)

    # creation of a rollerbearing instance
    rollBear = RollerBearing(samplingRate, estimateRpm, **bearingConfig)
    rollBear.getDefectFreq(estimateRpm)

    ########################bpfo detection ##############################################
    bpfoRes, defectFreqBpfo, scoreBpfo = myVib.detectBpfo(rollBear.ballPassOuterFreq, freqs, amps, \
                            percentDeviation = 1, nbStds = 1.5, percentPeakDeviation = 40, \
                            harmonics = 3, rangeFreq = int(len(freqs) / 10))

    ########################bpfi detection ##############################################
    # demodulation of the signal
    freqs1, amps1, timeAxis, enveloppe = myVib.autoCorrAndRedressWaveform(samplingRate,
waveform, \
                                                4 * rollBear.ballPassInnerFreq, orderFilt=1)

    bpfiRes, defectFreqBpfi, scoreBpfi = myVib.detectBpfo(rollBear.ballPassInnerFreq, freqs1[ : ],
amps1[ : ], \
                            percentDeviation = 5, nbStds = 1, percentPeakDeviation = 40, \
                            harmonics = 2, rangeFreq = int(len(freqs) / 10), minSize = 0.05)



    if bpfoRes :
        if df["defect"][k] == "no" or df["type"][k] == "bpfi" : score["type1Bpfo"] += 1

        # calculation of rpm theoritical
        rpmFound = getRpmBack(defectFreqBpfo, **bearingConfig)
```



```
        estimationRpmDic["bpfo"].append(rpmFound)
```

```
    elif bpfiRes:
        if df["defect"][k] == "no" or df["type"][k] == "bpfo" : score["type1Bpfi"] += 1

        # calculation of rpm theoritical
        rpmFound = getRpmBack(defectFreqBpfi, defectType = "bpfi", **bearingConfig)

        estimationRpmDic["bpfi"].append(rpmFound)

    else :
        if df["type"][k] == 'bpfo' : score["type2Bpfo"] += 1
        if df["type"][k] == 'bpfi' : score["type2Bpfi"] += 1




print("########False positive rate bpfo = ###########")
print(score["type1Bpfo"] / (score["totalNoDefect"] + score["totalBpfi"]))

print("########Missed spectrum bpfo = ##############")
print(score["type2Bpfo"] / score["totalBpfo"])

print("########False positive rate bpfi = ###########")
print(score["type1Bpfi"] / (score["totalNoDefect"] + score["totalBpfo"]))

print("########Missed spectrum bpfi= ##############")
print(score["type2Bpfi"] / score["totalBpfi"])
print(score)
print("########## estimation rpm with bpfo defect#################")
print(estimationRpmDic["bpfo"])
print("########## estimation rpm with bpfi defect#################")
print(estimationRpmDic["bpfi"])
```

```
########EXPERIMENT 2 : we have a partial knowledge of the rpm and the number of rolling elements############

import pandas as pd
# only one parameter is going to be modified : the amplitude of the defect
# ampli varies from .1 to 1
# let's create 3 signals per .1 of ampli (ie : 30 defect signals)

# the rpm is going to vary from + 300 to 500 rpm

# bearing parameters (coming from GP):

for rotSpeed in range(1, 8) : #
    rpm = rotSpeed * 50 + 250 #
    numElements = 20
    pitchDiam = 353
    ballDiam = 45
    contactAngle = 11



    bearingConfig = {"ballDiam" : ballDiam,
    "pitchDiam" : pitchDiam,
    "numElements" : numElements,
    "contactAngle" : contactAngle}

    # signal parameter
    samplingRate = 4096
    nbPts = samplingRate * 5

    # creation of a vibmachine instance
    myVib = VibMachine(samplingRate, rpm)
```

```
# get default frequencies
# creation of a rollerbearing instance
rollBear = RollerBearing(samplingRate, rpm, **bearingConfig)



rollBear.getDefectFreq(rpm)
freqBpfo = rollBear.ballPassOuterFreq
freqBpfi = rollBear.ballPassInnerFreq

bearingConfig = {"ballDiam" : ballDiam,
   "pitchDiam" : pitchDiam,
   "numElements" : numElements,
   "contactAngle" : contactAngle}

dicSignals = {"ampliImpact" : [], "data" : [], "defect" : [], "type" : []}

# bpfo data generation
for k in range(1, 10 + 1) :

   for j in range(3) :
      sign = genSign(samplingRate, nbPts, myVib)
      sign = genBpfo(sign, samplingRate, freqBpfo, dureeImpact = 10, ampliImpact = (0.3 + 0.1 * k))
      dicSignals["ampliImpact"].append(k * 0.1)
      dicSignals["data"].append(sign)
      dicSignals["defect"].append("yes")
      dicSignals["type"].append("bpfo")

# bpfi data generation
for k in range(1, 10 + 1) :
```

```
    for j in range(3) :

        sign = genSign(samplingRate, nbPts, myVib)
        sign = genBpfi(sign, samplingRate, freqBpfi, rpm, dureeImpact = 10, ampliImpact = 0.4 * k)
        dicSignals["ampliImpact"].append(k * 0.1)
        dicSignals["data"].append(sign)
        dicSignals["defect"].append("yes")
        dicSignals["type"].append("bpfi")



# now adding some signals with no defect
n = len(dicSignals["ampliImpact"])

# let's say only 25% of our data has a defect
for k in range(0 * n) :
    sign = genSign(samplingRate, nbPts, myVib)
    dicSignals["ampliImpact"].append(0)
    dicSignals["data"].append(sign)
    dicSignals["defect"].append('no')
    dicSignals["type"].append("None")

df = pd.DataFrame(dicSignals)

df.to_pickle("data.exp2.rpm=" + str(rotSpeed * 50 + 250) + ".pkl") #
```

```
# function looking for defects


def checkDefect(dataFrame, bearingConfig, score, estimationRpmDic, rpm = 300) :
   for k in range(n) :


      # generation of fft
      waveform = dataFrame['data'][k]


      freqs, amps = myVib.generateFFT(waveform)


      # to get same results
      np.random.seed(2)


      # generation of random rpm
      estimateRpm = rpm * (np.random.randint(-40, 41) / 100 + 1)



      # creation of a rollerbearing instance
      rollBear = RollerBearing(samplingRate, estimateRpm, **bearingConfig)
      rollBear.getDefectFreq(estimateRpm)



      ########################bpfo detection ##############################################
      bpfoRes = False
      if dataFrame["type"][k] != "bpfi":
         bpfoRes, defectFreqBpfo, scoreBpfo = myVib.detectBpfo(rollBear.ballPassOuterFreq, freqs,
amps, \
                              percentDeviation = 4, nbStds = 1.5, percentPeakDeviation = 40, \
                              harmonics = 3, rangeFreq = int(len(freqs) / 10))
```

```
    ########################bpfi detection ##############################################


    # demodulation of the signal

    freqs1, amps1, timeAxis, enveloppe = myVib.autoCorrAndRedressWaveform(samplingRate, waveform, \
                                                                          4 * rollBear.ballPassInnerFreq, orderFilt=1)


    bpfiRes, defectFreqBpfi, scoreBpfi = myVib.detectBpfo(rollBear.ballPassInnerFreq, freqs1[20 : ], amps1[20 : ], \
                                percentDeviation = 20, nbStds = 1, percentPeakDeviation = 40, \
                                harmonics = 2, rangeFreq = int(len(freqs) / 5), minSize = 0.05)




    if bpfoRes :


        if dataFrame["defect"][k] == "no" or dataFrame["type"][k] == "bpfi" : score["type1Bpfo"] += 1

        if dataFrame["type"][k] == "bpfi" : score["type2Bpfi"] += 1


        """

        plt.title("fils de puuuuuuute!m!!!")

        plt.plot(freqs[20 : 300] / rollBear.ballPassOuterFreq, 20 * amps[20 : 300])

        plt.show()

        """


        # calculation of rpm theoritical

        rpmFound = getRpmBack(defectFreqBpfo, **bearingConfig)


        estimationRpmDic["bpfo"].append(rpmFound)

    elif bpfiRes:

        if dataFrame["defect"][k] == "no" or dataFrame["type"][k] == "bpfo" : score["type1Bpfi"] += 1

        if dataFrame["type"][k] == "bpfo" : score["type2Bpfo"] += 1
```

```
            # calculation of rpm theoritical
            rpmFound = getRpmBack(defectFreqBpfi, defectType = "bpfi", **bearingConfig)

            estimationRpmDic["bpfi"].append(rpmFound)

        else :
            if dataFrame["type"][k] == 'bpfo' : score["type2Bpfo"] += 1
            if dataFrame["type"][k] == 'bpfi' : score["type2Bpfi"] += 1

            """
            plt.title(df["type"][k])
            plt.plot(freqs[20 : 300] / rollBear.ballPassInnerFreq, 20 * amps[20 : 300])
            plt.show()
            """

    return(score, estimationRpmDic)
```

```
##################### 3rd experiment : let's suppose pitch diam varies
#######################################################

# we suppose we are only going to work with one speed (100 rpm)



# reading the data

from ast import literal_eval

# now we have the data, we can start the experimentation part

df = pd.read_pickle("data.exp2.rpm=300.pkl")

n = len(df)




# brute force part

rpm = 300


for numElements in range(18, 23) :

   for pitch in range(335, 371, 5) :

      # type 1 = false positive , type2 = missed a bearing with a defect

      score = {"type1Bpfo" : 0, "type2Bpfo" : 0, "type1Bpfi" : 0, "type2Bpfi" : 0, \

            "totalNoDefect" : len(df[df["defect"] == "no"]), \

            "totalBpfo" : len(df[df["type"] == "bpfo"]), "totalBpfi" : len(df[df["type"] == "bpfi"])}


      estimationRpmDic = {"bpfo" : [], "bpfi" : []}



      bearingConfig = {"ballDiam" : 45,

         "pitchDiam" : pitch,

         "numElements" : numElements, # going to vary from +- 2 rolling elements (18 to 22)

         "contactAngle" : 11}
```

```
# creating function to look for defects
score, estimationRpmDic = checkDefect(df, bearingConfig, score, estimationRpmDic)

print("################### number of rolling elements = " + str(numElements) + \
    " pitch diam = " + str(pitch) + " ##################")
print("########False positive rate bpfo = ############")
print(score["type1Bpfo"] / (score["totalNoDefect"] + score["totalBpfi"]))

print("########Missed spectrum bpfo = ##############")
print(score["type2Bpfo"] / score["totalBpfo"])

print("########False positive rate bpfi = ############")
print(score["type1Bpfi"] / (score["totalNoDefect"] + score["totalBpfo"]))

print("########Missed spectrum bpfi= ##############")
print(score["type2Bpfi"] / score["totalBpfi"])

print(score)

print("########## estimation rpm with bpfo defect#################")
print(estimationRpmDic["bpfo"])

print("########## estimation rpm with bpfi defect#################")
print(estimationRpmDic["bpfi"])
```

REFERENCES

References

1. World Economic Forum (ed) (2018) The Fourth Industrial Revolution: what it means, how to respond

2. Monostori L (2014) Cyber-physical Production Systems: Roots, Expectations and R&D Challenges. Procedia CIRP 17: 9–13. doi: 10.1016/j.procir.2014.03.115

3. Paz NM, Leigh W (1994) Maintenance Scheduling: Issues, Results and Research Needs. Int Jrnl of Op & Prod Mnagemnt 14(8): 47–69. doi: 10.1108/01443579410067135

4. Robert Bond Randall (2011) Vibration-based Condition Monitoring. John Wiley & Sons

5. Great Britain (1979) A guide to the condition monitoring of machinery. H.M. Stationery Off, London

6. Ahmad R, Kamaruddin S (2012) An overview of time-based and condition-based maintenance in industrial application. Computers & Industrial Engineering 63(1): 135–149. doi: 10.1016/j.cie.2012.02.002

7. Grall A, Dieulle L, Berenguer C et al. (2002) Continuous-time predictive-maintenance scheduling for a deteriorating system. IEEE Trans. Rel. 51(2): 141–150. doi: 10.1109/TR.2002.1011518

8. Bloch HP, Geitner FK (2012) Machinery Failure Analysis and Troubleshooting: Practical Machinery Management for Process Plants. Elsevier Science & Technology Books, San Diego, CA, USA

9. Jardine AKS, Lin D, Banjevic D (2006) A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mechanical Systems and Signal Processing 20(7): 1483–1510. doi: 10.1016/j.ymssp.2005.09.012

10. Prajapati A, Bechtel J, Ganesan S (2012) Condition based maintenance: a survey. J of Qual in Maintenance Eng 18(4): 384–400. doi: 10.1108/13552511211281552

11. Lu B, Durocher D, Stemper P (2009) Predictive maintenance techniques. IEEE Ind. Appl. Mag. 15(6): 52–60. doi: 10.1109/MIAS.2009.934444

12. Tandon N, Choudhury A (1999) A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings. Tribology International 32(8): 469–480. doi: 10.1016/S0301-679X(99)00077-8

13. Shiroishi J, Li Y, Liang S et al. (1997) BEARING CONDITION DIAGNOSTICS VIA VIBRATION AND ACOUSTIC EMISSION MEASUREMENTS. Mechanical Systems and Signal Processing 11(5): 693–705. doi: 10.1006/mssp.1997.0113

14. Cooley JW, Tukey JW (1965) An algorithm for the machine calculation of complex Fourier series. Mathematics of Computation: 297–301

15. E. 0. Brigham, R. E. Morrow (1967) The fast Fourier transform. IEEE Spectrum 4: 63–70

16. Pierre Wickramarachi (2003) Effects of Windowing on the Spectral Content of a Signal. http://www.dataphysics.com/downloads/technical/Effects-of-Windowing-on-the-Spectral-Content-of-a-Signal-by-P.-Wickramarachi.pdf

17. Harris FJ (1978) On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform. Proceedings of the IEEE: 51–83

18. K. S. Brian P. Graney (2011) Rolling Element Bearing Analysis. The American Society for Nondestructive Testing 70: 78–85

19. N.Tandon (1994) A comparison of some vibration parameters A comparison for the condition monitoring of rolling element bearings Volume 12, Issue 3

20. H. F. Martin (1995) Application of Statistical Moments to Bearing Failure. Applied Acoustics: 66–77

21. Martin HR, Honarvar F (1995) Application of statistical moments to bearing failure detection. Applied Acoustics 44(1): 67–77. doi: 10.1016/0003-682X(94)P4420-B

22. F. Honarvar, H. R. Martin (1997) New Statistical Moments for Diagnostics of Rolling Element Bearings. Journal of Manufacturing Science and Engineering 199

23. N. C. Tandon (1997) An Analytical Model for the Prediction of the Vibration. Journal of Sound and Vibrations 205: 275–292

24. Montgomery D (2005) Introduction to Statistical Quality Control. John Wiley & Sons, Hoboken, New Jersey

25. Caballero Morales SO (2013) Economic Statistical Design of integrated X-bar-S control chart with Preventive Maintenance and general failure distribution. PLoS ONE 8(3): e59039. doi: 10.1371/journal.pone.0059039

26. NIST (2012) NIST/SEMATECH e-Handbook of Statistical Methods: 6.3.2.1. Shewhart X-bar and R and S Control Charts

27. Lloyd S. Nelson (1984) The Shewhart Control Chart—Tests for Special Causes. Journal of Quality Technology 16: 237–239

28. G. C. &. O. Griffiths (2010) The Probability of an Out of Control Signal fromNelson's Supplementary Zig-Zag Test. Journal of Statistical Theory and Practice: 609–615

29. B. P. Bogert, M. J. R. Healy, and J. W. Tukey (1963) The Quefrency Alanysis [sic] of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking 15: 209–243

30. Robert B. Randall (2017) A history of cepstrum analysis and its application to mechanical problems, Mechanical Systems and Signal Processing Volume 97: Pages 3-19

31. Tahar Fakhfakh, Fakher Chaari, Mohamed Haddar (2005) Numerical and experimental analysis of a gear system with teeth defects. International of Advanced Manufacturing Technologies 25: 542–550

32. E. Ruffio, D. Saury, D. Petit, M. Girault (2011) Tutorial 2: Zero-Order optimization algorithms. Eurotherm School METTI

33. MIT OpenCourseWare (2007) 18.02 Multivariable Calculus. https://ocw.mit.edu/courses/mathematics/18-02-multivariable-calculus-fall-2007/readings/plane_vector_fld.pdf. Accessed 04/17/2019

34. Wotao Yin (2015) OptimizationGradient descent. http://www.math.ucla.edu/~wotaoyin/math273a/slides/Lec3_gradient_descent_273a_2015_f.pdf. Accessed 04/17/2019

35. IEEE Neural Networks Society; IEEE International conference on fuzzy systems (1993) 1993 IEEE International Conference on Neural Networks, San Francisco, California, March 28-April 1, 1993. IEEE; IEEE Neural Networks Council; Available from IEEE Service Center, New York, Piscataway, NJ

36. Baldi P (1995) Gradient descent learning algorithm overview: a general dynamical systems perspective. IEEE Trans Neural Netw 6(1): 182–195. doi: 10.1109/72.363438

37. Ian Goodfellow, Yoshua Bengio, Aaron Courville (2016) Deep Learning. MIT Press

38. Pascanu R, Dauphin YN, Ganguli S et al. (5/19/2014) On the saddle point problem for non-convex optimization

39. Yinyu Ye Second Order Optimization Algorithms I. https://web.stanford.edu/class/msande311/lecture13.pdf. Accessed 04/17/2019

40. Robert M. Freund (2004) Newton's Method for Unconstrained Optimization. https://ocw.mit.edu/courses/sloan-school-of-management/15-084j-nonlinear-programming-spring-2004/lecture-notes/lec3_newton_mthd.pdf. Accessed 04/15/2019

41. Engineers Edge Angular contact bearing review. https://www.engineersedge.com/bearing/angular_contact_bearings.htm. Accessed 04/24/2019

42. Angular contact bearings. https://www.bocabearings.com/products/bearing-and-ball-types/angular-contact-bearings. Accessed 04/24/2019